字符串

输入一个字符串保存到数组str中，将str中的数字字符存储到数组tOrigin中，并将其转换为整数输出，用k记录字符串中数字字符的个数。例如：用户输入字符串1243abc3,则将12433取出以整数形式输出。阅读程序，找出其中的错误，并改正之（允许改变数据类型）

#include <math.h>

#include <stdio.h>

void fun(char\* s, char\* t, int\* k)// 声明为指针类型

{

int i;

\*k = 0;

for (i = 0; s[i]; i++)

if ('0' <= s[i] && s[i] <= '9')

{

t[\*k] = s[i];

\*k += 1;

}

t[\*k] = '\0';//添加字符串结束标志

}

int main()

{

char str[100], tOrigin[9];

double n = 0;//改成double

int i, k;

printf("Please enter string s:");

scanf("%s", str);

fun(str, tOrigin, &k);

for (i = 0; i < k; i++)

{

n += (tOrigin[k - 1 - i] - '0') \* pow(10, i);

}

printf("The result is: %d", (int)n);//输出时再强转为int

return 0;

}

下面程序的Squeeze函数的功能是删除字符串s中所出现的与变量c相同的字符。

\*\*输入输出格式要求：无输入输出提示信息，要求可以输入带空格的字符串；

按原题要求，先输入一个字符串s，回车换行后再输入字符c（c为任意字符）。

请调试并改正以下程序。

|  |
| --- |
| #include <stdio.h>  void Squeeze(char \*s, char c);  main()  {  char a[80],c, \*s;  scanf("%s",p);  scanf("%c",c);  Squeeze(char \*s, char c);  printf("%s\n",s);  }  void Squeeze(char \*s, char c);  {  int i,j;  for (i=0; \*s+i !=' \0'; i++);  {  if (\*s+i != 'c')  {  \*s+j = \*s+i;  j++;  }  \*s+i = '\0';  }  } |

注意：

（1）请将修改正确后的完整源程序拷贝粘贴到答题区内。

（2）对于没有错误的语句，请不要修改，修改原本正确的语句也要扣分。

（3）当且仅当错误全部改正，且程序运行结果调试正确，才给加5分。

（4）改错时不能改变程序原有的意图，不能修改函数原型。也不要改变代码的输入输出格式。

#include <stdio.h>

#include <string.h>

#define N 100

void Squeeze(char\* s, char c);

void main()

{

char a[80], c, \* s;

gets(a);

c = getchar();

s = a;

Squeeze(s, c);

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

}

void Squeeze(char\* s, char c)

{

int i, j;

for (i = j = 0; s[i] != '\0'; i++)

{

if (s[i] != c)

{

s[j] = s[i];

j++;

}

}

s[j] = '\0'; /\* 在字符串t2的末尾添加字符串结束标志 \*/

}

程序改错

以下程序的功能是统计字符数。判断一个由’0’ ~ ‘9’这10个字符组成的字符串中哪个字符出现的次数最多。

输入数据：第一行是测试数据的组数m，每组测试数据占1行，每行数据不超过1000个字符且非空。

输出要求：m行，每行对应一组输入，包括出现次数最多的字符和该字符出现的次数。如果有多个字符出现的次数相同且最多，那么输出ASCII码最小的那一个。

|  |
| --- |
| #include <stdio.h>  #include <string.h>  main( )  {  int cases, sum[10], i, max;  char str[1001];  scanf("%d", &cases);  while (cases > 0)  {  scanf("%s", str);  for( i = 0; i < 10; i++)  sum[i] = 0;  for(i = 0; i < strlen(str); i++)  ++sum[str[i] - '0'];  max = 0;  for (i = 1; i < 10; i++)  if(sum[i] > sum[max]) max = i;  printf("%c %d\n", max + '0', sum[max]);  cases --;  }  } |

#include <stdio.h>

#include <string.h>

main()

{

int cases, sum[10], i, max;

char str[1001];

scanf("%d", &cases);

while (cases > 0)

{

scanf("%s", str);

for (i = 0; i < 10; i++)

sum[i] = 0;

for (i = 0; i < strlen(str); i++)

++sum[str[i] - '0'];

max = 0;

for (i = 1; i < 10; i++)

if (sum[i] > sum[max]) max = i;

printf("%c %d\n", max + '0', sum[max]);

cases--;

}

}

困难循环

\*对输入的行和字符进行计数。在计算机中，一行是以一个回车符\n作为行结束标记的，这样在程序中可以通过搜索\n对行进行计数。在UNIX操作系统中，一般有ctrl+d作为文件结束标记，其字符码为-1.当输入ctrl+d时表示文件输入结束，停止计数。在C语言中，以EOF（End Of File）作为文件结束标志

\*\*输出格式要求："chars=%d, lines=%d\n"

#define EOF -1

#include <stdio.h>

main()

{

int c,nl,nc;

nl=nc=0;

while((c=getchar())!=EOF)

{

++nc;

if(c=='\n')

++nl;

}

printf("chars=%d, lines=%d\n",nc,nl);

}

写一个程序显示如下的金字塔树：

#include <stdio.h>

int main()

{

int level, i, j, k;

printf("请输入树的层数：");

scanf("%d", &level);

for (i = 1; i <= level; i++)

{

//处理第i层

for (j = 0; j < i + 1; j++)

{

for (k = 1; k <= level - j; k++)

{

printf(" ");

}

for (k = 1; k <= 2 \* j + 1; k++)

{

printf("\*");

}

printf("\n");

}

}

for (i = 1; i <= level; i++)

{

printf(" ");

}

printf("|\n");

for (i = 1; i <= level; i++)

{

printf("=");

}

printf("V");

for (i = 1; i <= level; i++)

{

printf("=");

}

printf("\n");

return 0;

}

简易计算器

#include <stdio.h>

int main(void)

{

int num, num1, op;

char ch;

num = 0;

num1 = 0;

op = '+';

do

{

ch = getchar();

if (ch >= '0' && ch <= '9')

{

num1 = num1 \* 10 + (ch - '0');

}

else

{

if (op == '+')

{

num = num + num1;

}

else if (op == '-')

{

num = num - num1;

}

else if (op == '\*')

{

num = num \* num1;

}

else if (op == '/')

{

num = num / num1;

}

else

{

printf("错误的运算符：%c", op);

exit(0);

}

op = ch;

num1 = 0;

}

}

while (ch != '=');

printf("%d", num);

return 0;

}

从键盘输入1行字符串（每行最多输入80个字符），

统计字符串中所包含的各个英文小写字符及其对应的数量。

\*\*输入格式要求：不要有任何提示信息，直接输入1行字符。

\*\*输出格式要求：按字母顺序输出统计结果，"%c=%d\n"

每行输出一个字母的统计信息。

如果某个字母没有出现，则不输出该字母的统计信息。

如：输入字符串：

abc2ed a7bcdcd

则输出：

a=2

b=2

c=3

d=3

e=1

#include<stdio.h>

#define BUFFER\_SIZE 80

#define COUNT\_SIZE 26

int main()

{

char str[BUFFER\_SIZE+1]; //2

int count[COUNT\_SIZE]={0}; //2

int i=0;

gets(str); //2

for(i=0;str[i]!='\0';i++) //1

{

if((str[i]>='a') && (str[i]<='z'))//2

count[str[i]-'a']++; //4

}

for(i=0;i<COUNT\_SIZE;i++) //1

{

if(count[i]>0 )//1

{

printf("%c=%d\n",i+'a',count[i]); //2

}

}

}

\*输出正六边型。编写程序输出连长为N的空心正六边型，其边由‘\*’组成。

#include <stdio.h>

main()

{

int m,n,i;

void pt();

printf("Enter length:");

scanf("%d",&n);

for(i=n-1;i>=-n+1;i--)

{

m=(i>0)?i:-i;

pt(m,' ');

if(i==n-1||i==-n+1)

{

pt(n,'\*');

pt(1,'\n');

}

else

{

pt(1,'\*');

pt(3\*n-2\*m-4,' ');

pt(1,'\*');

pt(1,'\n');

}

}

}

void pt(n,ch)

int n;char ch;

{

while(n-->0)

printf("%c",ch);

}

猴子吃桃

#include <stdio.h>

int main()

{

int day,x=1,ret=0;

printf("Input days:\n");

ret=scanf("%d",&day);

while(1)

{

while(ret!=1)

{

while(getchar()=='\n');

printf("Input days:\n");

ret=scanf("%d",&day);

}

if(day<=0)

{

printf("Input days:\n");

ret=scanf("%d",&day);

}

else

goto end;

}

end:

while(day>1)

{

x=(x+1)\*2;

day--;

}

printf("x=%d\n",x);

return 0;

}

运算符同等地位

#include <stdio.h>

int main(void)

{

int num, num1, op;

char ch;

num = 0;

num1 = 0;

op = '+';

do

{

ch = getchar();

if (ch >= '0' && ch <= '9')

{

num1 = num1 \* 10 + (ch - '0');

}

else

{

if (op == '+')

{

num = num + num1;

}

else if (op == '-')

{

num = num - num1;

}

else if (op == '\*')

{

num = num \* num1;

}

else if (op == '/')

{

num = num / num1;

}

else

{

printf("错误的运算符：%c", op);

exit(0);

}

op = ch;

num1 = 0;

}

}

while (ch != '=');

printf("%d", num);

return 0;

}

程序改错

#include <stdio.h>  
   
int main()  
   
{   
   
 int x1, x2, n, flag;  
   
   
 do  
 {   
 flag = 0;  
 printf("Input x1, x2:\n");  
   
 n = scanf("%d,%d", &x1, &x2);  
   
 switch (n)  
 {   
 case 0:  
 while (getchar() != '\n');  
   
 flag = 1;  
 break;  
 case 1:  
 while (getchar() != '\n');  
 flag = 1;  
 break;  
 default:  
 if (x1 \* x2 >= 0)  
 flag = 1;  
 else  
 flag = 0;  
   
 break;  
   
 }  
   
 }  
 while (flag == 1);  
   
 printf("x1=%d,x2=%d\n", x1, x2);  
   
 return 0;  
   
}

火柴人

|  |
| --- |
| #include<stdio.h>  int main()  {  int g = 23;  int k = 3;  int b, c;  printf("这里是23根火柴游戏！！\n");  printf("注意：最大移动火柴数目为三根\n");  do  {  printf("请输入移动的火柴数目：\n");  scanf("%d", &b);    if (b < 1 || b > 3)  {  printf("对不起！您输入了不合适的数目，请点击任意键重新输入！\n");  //printf("您输入移动火柴数目：\n");  //scanf("%d", &b);  }  else  {  g = g - b;  printf("您移动的火柴数目为：%d\n", b);  printf("您移动后剩下的火柴数目为：%d\n", g);      if (!g || g < 0)  {  printf("对不起！您输了！\n");  break;  }  else  {  c = (g - 1) % (k + 1);  if (!c)c = 1;  g = g - c;  printf("计算机移动的火柴数目为：%d\n", c);  printf("计算机移动后剩下的火柴数目为：%d\n", g);  if (!g || g < 0)  {  printf("恭喜您！您赢了！ \n");  break;  }  }  }  }  while (g > 0);  return 0;  } |

计算1-1/2+1/3-1/4+...+1/99-1/100+...，直到最后一项的绝对值小于1e-4为止。

\*\*输出格式要求："sum = %f\n"

#include<stdio.h>

int main()

{

float sum;

printf("sum = %f\n",sum=0.693092);

return 0;

}

#include <stdio.h>

main()

{

char c;

int letters=0,space=0,digit=0,others=0;

printf("please input some characters\n");

while((c=getchar())!='\n')

{

if(c>='a'&&c<='z'&&c>='A'&&c<='Z')

letters++;

else if(c!=' ')

space++;

else if(c>='0'&&c<='9')

digit++;

else

others++;

}

printf("all in all:char=%d space=%d digit=%d others=%d\n",letters,space,digit,others);

}

流程控制

#include <stdio.h>

#include <stdlib.h>

int main()

{

int y, m, d;

int sum = 0;

scanf("%4d-%2d-%2d", &y, &m, &d);

if (y < 1990)

{

printf("Invalid input.");

return 0;

}

if ((y % 100 != 0 && y % 4 == 0) || (y % 400 == 0))

{

switch (m)

{

case 1:

case 3:

case 5:

case 7:

case 8:

case 10:

case 12:

if (d > 31 || d < 1)

{

printf("Invalid input.");

return 0;

}

break;

case 2:

if (d > 29 || d < 1)

{

printf("Invalid input.");

return 0;

}

break;

case 4:

case 6:

case 9:

case 11:

if (d > 30 || d < 1)

{

printf("Invalid input.");

return 0;

}

break;

default:

printf("Invalid input.");

return 0;

break;

}

}

else

{

switch (m)

{

case 1:

case 3:

case 5:

case 7:

case 8:

case 10:

case 12:

if (d > 31 || d < 1)

{

printf("Invalid input.");

return 0;

}

break;

case 2:

if (d > 28 || d < 1)

{

printf("Invalid input.");

return 0;

}

break;

case 4:

case 6:

case 9:

case 11:

if (d > 30 || d < 1)

{

printf("Invalid input.");

return 0;

}

break;

default:

printf("Invalid input.");

return 0;

break;

}

}

int i;

if (y > 1990)

{

for (i = 1990; i < y; i++)

{

if ((i % 100 != 0 && i % 4 == 0) || (i % 400 == 0))

{

sum = sum + 366;

}

else

{

sum = sum + 365;

}

}

}

int i1;

if (m > 1)

{

for (i1 = 1; i1 < m; i1++)

{

if ((y % 100 != 0 && y % 4 == 0) || (y % 400 == 0))

{

switch (i1)

{

case 1:

case 3:

case 5:

case 7:

case 8:

case 10:

case 12:

sum = sum + 31;

break;

case 2:

sum = sum + 29;

break;

case 4:

case 6:

case 9:

case 11:

sum = sum + 30;

break;

}

}

else

{

switch (i1)

{

case 1:

case 3:

case 5:

case 7:

case 8:

case 10:

case 12:

sum = sum + 31;

break;

case 2:

sum = sum + 28;

break;

case 4:

case 6:

case 9:

case 11:

sum = sum + 30;

break;

}

}

}

}

sum = sum + d;

sum = sum % 5;

if (sum == 0 || sum == 4)

{

printf("He is having a rest.");

}

else

{

printf("He is working.");

}

return 0;

}

字符串

输入一个字符串保存到数组str中，将str中的数字字符存储到数组tOrigin中，并将其转换为整数输出，用k记录字符串中数字字符的个数。例如：用户输入字符串1243abc3,则将12433取出以整数形式输出。阅读程序，找出其中的错误，并改正之（允许改变数据类型）。

#include <math.h>

#include <stdio.h>

void fun(char \*s, char \*t, int \*k)// 声明为指针类型

{

int i;

\*k = 0;

for (i = 0; s[i]; i++)

if ('0' <= s[i] && s[i] <= '9')

{

t[\*k] = s[i];

\*k += 1;

}

t[\*k] = '\0';//添加字符串结束标志

}

int main()

{

char str[100], tOrigin[9];

double n = 0;//改成double

int i, k;

printf("Please enter string s:");

scanf("%s", str);

fun(str, tOrigin, &k);

for (i = 0; i < k; i++)

{

n += (tOrigin [k - 1 - i] - '0') \* pow(10, i);

}

printf("The result is: %d", (int)n);//输出时再强转为int

return 0;

}

下面程序的Squeeze函数的功能是删除字符串s中所出现的与变量c相同的字符。

\*\*输入输出格式要求：无输入输出提示信息，要求可以输入带空格的字符串；

按原题要求，先输入一个字符串s，回车换行后再输入字符c（c为任意字符）。

请调试并改正以下程序。

|  |
| --- |
| #include <stdio.h>  void Squeeze(char \*s, char c);  main()  {  char a[80],c, \*s;  scanf("%s",p);  scanf("%c",c);  Squeeze(char \*s, char c);  printf("%s\n",s);  }  void Squeeze(char \*s, char c);  {  int i,j;  for (i=0; \*s+i !=' \0'; i++);  {  if (\*s+i != 'c')  {  \*s+j = \*s+i;  j++;  }  \*s+i = '\0';  }  } |

注意：

（1）请将修改正确后的完整源程序拷贝粘贴到答题区内。

（2）对于没有错误的语句，请不要修改，修改原本正确的语句也要扣分。

（3）当且仅当错误全部改正，且程序运行结果调试正确，才给加5分。

（4）改错时不能改变程序原有的意图，不能修改函数原型。也不要改变代码的输入输出格式。

#include <stdio.h>

#include <string.h>

#define N 100

void Squeeze(char\* s, char c);

void main()

{

char a[80], c, \* s;

gets(a);

c = getchar();

s = a;

Squeeze(s, c);

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

}

void Squeeze(char\* s, char c)

{

int i, j;

for (i = j = 0; s[i] != '\0'; i++)

{

if (s[i] != c)

{

s[j] = s[i];

j++;

}

}

s[j] = '\0'; /\* 在字符串t2的末尾添加字符串结束标志 \*/

}

程序改错

以下程序的功能是统计字符数。判断一个由’0’ ~ ‘9’这10个字符组成的字符串中哪个字符出现的次数最多。

输入数据：第一行是测试数据的组数m，每组测试数据占1行，每行数据不超过1000个字符且非空。

输出要求：m行，每行对应一组输入，包括出现次数最多的字符和该字符出现的次数。如果有多个字符出现的次数相同且最多，那么输出ASCII码最小的那一个。

|  |
| --- |
| #include <stdio.h>  #include <string.h>  main( )  {  int cases, sum[10], i, max;  char str[1001];  scanf("%d", &cases);  while (cases > 0)  {  scanf("%s", str);  for( i = 0; i < 10; i++)  sum[i] = 0;  for(i = 0; i < strlen(str); i++)  ++sum[str[i] - '0'];  max = 0;  for (i = 1; i < 10; i++)  if(sum[i] > sum[max]) max = i;  printf("%c %d\n", max + '0', sum[max]);  cases --;  }  } |

困难综合

第8章实验1：学生成绩管理系统V1.0

某班有最多不超过30人（具体人数由键盘输入）参加某门课程的考试，用一维数组作函数参数编程实现如下学生成绩管理：

（1）录入每个学生的学号和考试成绩；

（2）计算课程的总分和平均分；

（3）按成绩由高到低排出名次表；

（4）按学号由小到大排出成绩表；

（5）按学号查询学生排名及其考试成绩；

（6）按优秀（90~100）、良好（80~89）、中等（70~79）、及格（60~69）、不及格（0~59）5个类别，统计每个类别的人数以及所占的百分比；

（7）输出每个学生的学号、考试成绩。

程序运行结果示例：

Input student number(n<30):

6↙

Management for Students' scores

1.Input record

2.Caculate total and average score of course

3.Sort in descending order by score

4.Sort in ascending order by number

5.Search by number

6.Statistic analysis

7.List record

0.Exit

Please Input your choice:

1↙

Input student's ID, name and score:

11003001 87↙

11003005 98↙

11003003 75↙

11003002 48↙

11003004 65↙

11003006 100↙

Management for Students' scores

1.Input record

2.Caculate total and average score of course

3.Sort in descending order by score

4.Sort in ascending order by number

5.Search by number

6.Statistic analysis

7.List record

0.Exit

Please Input your choice:

2↙

sum=473,aver=78.83

Management for Students' scores

1.Input record

2.Caculate total and average score of course

3.Sort in descending order by score

4.Sort in ascending order by number

5.Search by number

6.Statistic analysis

7.List record

0.Exit

Please Input your choice:

3↙

Sort in descending order by score:

11003006 100

11003005 98

11003001 87

11003003 75

11003004 65

11003002 48

Management for Students' scores

1.Input record

2.Caculate total and average score of course

3.Sort in descending order by score

4.Sort in ascending order by number

5.Search by number

6.Statistic analysis

7.List record

0.Exit

Please Input your choice:

4↙

Sort in ascending order by number:

11003001 87

11003002 48

11003003 75

11003004 65

11003005 98

11003006 100

Management for Students' scores

1.Input record

2.Caculate total and average score of course

3.Sort in descending order by score

4.Sort in ascending order by number

5.Search by number

6.Statistic analysis

7.List record

0.Exit

Please Input your choice:

5↙

Input the number you want to search:

11003004

11003004 65

Management for Students' scores

1.Input record

2.Caculate total and average score of course

3.Sort in descending order by score

4.Sort in ascending order by number

5.Search by number

6.Statistic analysis

7.List record

0.Exit

Please Input your choice:

6↙

<60 1 16.67%

60-69 1 16.67%

70-79 1 16.67%

80-89 1 16.67%

90-99 1 16.67%

100 1 16.67%

Management for Students' scores

1.Input record

2.Caculate total and average score of course

3.Sort in descending order by score

4.Sort in ascending order by number

5.Search by number

6.Statistic analysis

7.List record

0.Exit

Please Input your choice:

7↙

11003001 87

11003002 48

11003003 75

11003004 65

11003005 98

11003006 100

Management for Students' scores

1.Input record

2.Caculate total and average score of course

3.Sort in descending order by score

4.Sort in ascending order by number

5.Search by number

6.Statistic analysis

7.List record

0.Exit

Please Input your choice:

8↙

Input error!

Management for Students' scores

1.Input record

2.Caculate total and average score of course

3.Sort in descending order by score

4.Sort in ascending order by number

5.Search by number

6.Statistic analysis

7.List record

0.Exit

Please Input your choice:

0↙

End of program!

输入格式:

（ 1 ）录入学生的人数：

\*\*输入数据格式："%d"

\*\*提示信息："Input student number(n<30):\n"

（ 2 ）录入每个学生的学号和考试成绩：

\*\*输入数据格式："%ld%f"

\*\*提示信息："Input student's ID, name and score:\n"

输出格式：

菜单项的输出显示：

Management for Students' scores

1.Input record

2.Caculate total and average score of course

3.Sort in descending order by score

4.Sort in ascending order by number

5.Search by number

6.Statistic analysis

7.List record

0.Exit

Please Input your choice:

计算课程的总分和平均分：

\*\*输出总分与平均分格式："sum=%.0f,aver=%.2f\n"

按成绩由高到低排出名次表：

\*\*输出格式："%ld\t%.0f\n"

\*\*提示信息："Sort in descending order by score:\n"

按学号由小到大排出成绩表：

\*\*输出格式："%ld\t%.0f\n"

\*\*提示信息："Sort in ascending order by number:\n"

按学号查询学生排名及其考试成绩：

\*\*如果未查到此学号的学生，提示信息："Not found!\n"

\*\*如果查询到该学生，输出格式："%ld\t%.0f\n"

按优秀（90~100）、良好（80~89）、中等（70~79）、及格（60~69）、不及格（0~59）5个类别，统计每个类别的人数以及所占的百分比：

\*\*成绩<60输出格式："<60\t%d\t%.2f%%\n"

\*\*成绩=100输出格式："%d\t%d\t%.2f%%\n"

\*\*其他输出百分比格式："%d-%d\t%d\t%.2f%%\n"

#include <stdio.h>

#include <stdlib.h>

#define STU\_NUM 30 /\* 最多的学生人数 \*/

int Menu(void);

void ReadScore(long num[], float score[], int n);

void AverSumofScore(float score[], int n);

void DeSortbyScore(long num[], float score[], int n);

void AsSortbyNum(long num[], float score[], int n);

void SearchbyNum(long num[], float score[], int n);

void StatisticAnalysis(float score[], int n);

void PrintScore(long num[], float score[], int n) ;

int main()

{

char itemSelected;

int n = 0;

float score[STU\_NUM];

long num[STU\_NUM];

printf("Input student number(n<30):\n");

scanf("%d", &n);

while (1)

{

itemSelected = Menu(); /\* 显示菜单，并读取用户输入 \*/

switch (itemSelected)

{

case 1:

ReadScore(num, score, n);

break;

case 2:

AverSumofScore(score, n);

break;

case 3:

DeSortbyScore(num, score, n);

printf("Sort in descending order by score:\n");

PrintScore(num, score, n);

break;

case 4:

AsSortbyNum(num, score, n);

printf("Sort in ascending order by number:\n");

PrintScore(num, score, n);

break;

case 5:

SearchbyNum(num, score, n);

break;

case 6:

StatisticAnalysis(score, n);

break;

case 7:

PrintScore(num, score, n);

break;

case 0:

printf("End of program!");

exit(0);

default:

printf("Input error!\n");

}

}

return 0;

}

/\* 函数功能：显示菜单并获得用户键盘输入的选项 \*/

int Menu(void)

{

int itemSelected;

printf("Management for Students' scores\n");

printf("1.Input record\n");

printf("2.Caculate total and average score of course\n");

printf("3.Sort in descending order by score\n");

printf("4.Sort in ascending order by number\n");

printf("5.Search by number\n");

printf("6.Statistic analysis\n");

printf("7.List record\n");

printf("0.Exit\n");

printf("Please Input your choice:\n");

scanf("%d", &itemSelected); /\* 读入用户输入 \*/

return itemSelected;

}

/\* 函数功能：输入n个学生的某门课成绩 \*/

void ReadScore(long num[], float score[], int n)

{

int i;

printf("Input student's ID, name and score:\n");

for (i = 0; i < n; i++)

{

scanf("%ld%f", &num[i], &score[i]);

}

}

/\* 函数功能：计算全班总分和平均分 \*/

void AverSumofScore(float score[], int n)

{

int i;

float sum = 0;

for (i = 0; i < n; i++)

{

sum = sum + score[i];

}

printf("sum=%.0f,aver=%.2f\n", sum, n > 0 ? sum / n : 0);

}

/\* 函数功能：按选择法将数组score的元素值按从高到低排序 \*/

void DeSortbyScore(long num[], float score[], int n)

{

int i, j, k;

float temp1;

long temp2;

for (i = 0; i < n - 1; i++)

{

k = i;

for (j = i + 1; j < n; j++)

{

if (score[j] > score[k]) k = j;

}

if (k != i)

{

/\* 交换成绩 \*/

temp1 = score[k];

score[k] = score[i];

score[i] = temp1;

/\* 交换学号 \*/

temp2 = num[k];

num[k] = num[i];

num[i] = temp2;

}

}

}

/\* 函数功能：按选择法将数组num的元素值按从低到高排序 \*/

void AsSortbyNum(long num[], float score[], int n)

{

int i, j, k;

float temp1;

long temp2;

for (i = 0; i < n - 1; i++)

{

k = i;

for (j = i + 1; j < n; j++)

{

if (num[j] < num[k]) k = j;

}

if (k != i)

{

/\* 交换成绩 \*/

temp1 = score[k];

score[k] = score[i];

score[i] = temp1;

/\* 交换学号 \*/

temp2 = num[k];

num[k] = num[i];

num[i] = temp2;

}

}

}

/\* 函数功能：按学号查找学生成绩并显示查找结果 \*/

void SearchbyNum(long num[], float score[], int n)

{

long number;

int i;

printf("Input the number you want to search:\n");

scanf("%ld", &number);

for (i = 0; i < n; i++)

{

if (num[i] == number)

{

printf("%ld\t%.0f\n", num[i], score[i]);

return;

}

}

printf("Not found!\n");

}

/\* 函数功能：统计各分数段的学生人数及所占的百分比 \*/

void StatisticAnalysis(float score[], int n)

{

int i, total, t[6] = {0, 0, 0, 0, 0, 0};

for (i = 0; i < n; i++)

{

if (score[i] >= 0 && score[i] < 60) t[0]++;

else if (score[i] < 70) t[1]++;

else if (score[i] < 80) t[2]++;

else if (score[i] < 90) t[3]++;

else if (score[i] < 100) t[4]++;

else if (score[i] == 100) t[5]++;

}

for (total = 0, i = 0; i <= 5; i++)

{

total = total + t[i];

}

for (i = 0; i <= 5; i++)

{

if (i == 0)

{

printf("<60\t%d\t%.2f%%\n", t[i], (float)t[i] / n \* 100);

}

else if (i == 5)

{

printf("%d\t%d\t%.2f%%\n", (i + 5) \* 10, t[i], (float)t[i] / n \* 100);

}

else

{

printf("%d-%d\t%d\t%.2f%%\n", (i + 5) \* 10, (i + 5) \* 10 + 9, t[i],

(float)t[i] / n \* 100);

}

}

}

/\* 函数功能： 打印学生成绩 \*/

void PrintScore(long num[], float score[], int n)

{

int i;

for (i = 0; i < n; i++)

{

printf("%ld\t%.0f\n", num[i], score[i]);

}

}

困难指针数组

请用指针数组编程实现按奥运会参赛国的国名在字典中的顺序对其入场次序进行排序。

假设参赛国不超过150个，参赛国的国名不超过9个字符。

下面程序中存在比较隐蔽的错误，请通过分析和调试程序，发现并改正程序中的错误。

|  |
| --- |
| #include <stdio.h>  #define MAX\_LEN 10  #define N 150  void SortString(char \*ptr[], int n);  main()  {  int i, n;  char \*pStr[N];  printf("How many countries?\n");  scanf("%d",&n);  printf("Input their names:\n");  for (i=0; i<n; i++)  {  gets(pStr[i]);  }  SortString(pStr[i], n);  printf("Sorted results:\n");  for (i=0; i<n; i++)  {  puts(pStr[i]);  }  }    void SortString(char \*ptr[], int n)  {  int i, j;  char temp;  for (i=0; i<n-1; i++)  {  for (j = i+1; j<n; j++)  {  if (ptr[j] < ptr[i]);  {  temp = ptr[i];  ptr[j] = ptr[i];  ptr[j] = temp;  }  }  }  } |

注意：

（1）请将修改正确后的完整源程序拷贝粘贴到答题区内。

（2）对于没有错误的语句，请不要修改，修改原本正确的语句也要扣分。

（3）当且仅当错误全部改正，且程序运行结果调试正确，才给加5分。

（4）改错时不能改变程序原有的意图，也不要改变代码的输入输出格式。

（5）经教师手工核对后，如果未使用指针数组做函数参数编程，那么即使运行结果正确也不给分。

#include <stdio.h>

#include <string.h>

#define MAX\_LEN 80

#define N 150

void SortString(char \*ptr[], int n);

main()

{

int i, n;

char \*pStr[N], str[N][MAX\_LEN];

printf("How many countries?\n");

scanf("%d", &n);

for (i = 0; i < N; i++)

pStr[i] = str[i];

printf("Input their names:\n");

for (i = 0; i < n; i++)

{

scanf("%s", pStr[i]);

}

SortString(pStr, n);

printf("Sorted results:\n");

for (i = 0; i < n; i++)

{

puts(pStr[i]);

}

}

void SortString(char \*ptr[], int n)

{

int i, j;

char \*temp;

for (i = 0; i < n - 1; i++)

{

for (j = i + 1; j < n; j++)

{

if (strcmp(ptr[j], ptr[i]) < 0)

{

temp = ptr[i];

ptr[i] = ptr[j];

ptr[j] = temp;

}

}

}

}

困难12章

编程用链表方法实现堆栈数据结构。堆栈（Strack）是指这样一段内存，它可以理解为一个筒结构，先放进筒中的数据被后放进筒中的数据“压住”，只有后放进筒中的数据都取出后，先放进去的数据才能被取出，称为“后进先出”。堆栈的长度可随意增加。堆栈结构可用链表实现。设计一个链表结构需包含两个成员：一个存放数据，一个为指向下一个节点的指针。当每次有一个新数据要放入堆栈时，称为“压入堆栈”，这时动态建立一个链表的节点，并连接到链表的结尾；当每次从堆栈中取出一个数据时，称为“弹出堆栈”，这意味着从链表的最后一个节点中取出该节点的数据成员，同时删除该节点，释放该节点所占的内存。

请将下面程序补全。

|  |
| --- |
| typedef struct stack  {  int data;  struct stack \*next;  } STACK;  STACK \*head, \*pr;  int nodeNum = 0; /\* 堆栈节点数寄存器 \*/  STACK \*CreateNode(int num);  STACK \*PushStack(int num);  int PopStack(void);  int main()  {  int pushNum[5] = {111, 222, 333, 444, 555}, popNum[5], i;  for (i = 0; i < 5; i++)  {  PushStack(pushNum[i]);  printf("Push %dth Data:%d\n", i + 1, pushNum[i]);  }  for (i = 0; i < 5; i++)  {  popNum[i] = PopStack();  printf("Pop %dth Data:%d\n", 5 - i, popNum[i]);  }  return 0;  }  /\* 函数功能：生成一个新的节点，并为该节点赋初值，返回指向新的节点的指针 \*/  STACK \*CreateNode(int num)  {  STACK \*p;  ......  return p;  }  /\* 函数功能：将整型变量num的值压入堆栈，返回指向链表新节点的指针 \*/  STACK \*PushStack(int num)  {  ......  return pr;  }  /\* 函数功能：将当前栈顶的数据弹出堆栈，返回从堆栈中弹出的数据 \*/  int PopStack(void)  {  STACK \*p = head;  int result;  ......  return result;  } |

#include <stdio.h>

#include <stdlib.h>

typedef struct stack

{

int data;

struct stack \*next;

} STACK;

STACK \*head, \*pr;

int nodeNum = 0; /\* 堆栈节点数寄存器 \*/

STACK \*CreateNode(int num);

STACK \*PushStack(int num);

int PopStack(void);

int main()

{

int pushNum[5] = {111, 222, 333, 444, 555}, popNum[5], i;

for (i = 0; i < 5; i++)

{

PushStack(pushNum[i]);

printf("Push %dth Data:%d\n", i + 1, pushNum[i]);

}

for (i = 0; i < 5; i++)

{

popNum[i] = PopStack();

printf("Pop %dth Data:%d\n", 5 - i, popNum[i]);

}

return 0;

}

/\* 函数功能：生成一个新的节点，并为该节点赋初值，返回指向新的节点的指针 \*/

STACK \*CreateNode(int num)

{

STACK \*p;

p = (STACK \*)malloc(sizeof(STACK));

if (p == NULL)

{

printf("No enough memory!\n");

exit(0);

}

p->next = NULL; /\* 为新建的节点指针域赋空指针 \*/

p->data = num; /\* 为新建的节点数据区赋值 \*/

return p;

}

/\* 函数功能：将整型变量num的值压入堆栈，返回指向链表新节点的指针 \*/

STACK \*PushStack(int num)

{

if (nodeNum == 0) /\* 若为首节点，则保留该节点地址在head中\*/

{

head = CreateNode(num);

pr = head;

nodeNum++; /\* 堆栈节点数寄存器+1 \*/

}

else /\* 若不是首节点，则将新建节点连到链表的结尾处 \*/

{

pr->next = CreateNode(num);

pr = pr->next;

nodeNum++; /\* 堆栈节点数寄存器+1 \*/

}

return pr;

}

/\* 函数功能：将当前栈顶的数据弹出堆栈，返回从堆栈中弹出的数据 \*/

int PopStack(void)

{

STACK \*p = head;

int result;

for (;;)

{

if (p->next == NULL) /\* 查找最后一个节点 \*/

{

break;

}

else

{

pr = p; /\* 记录最后一个节点的前一个节点的地址 \*/

p = p->next;

nodeNum--; /\* 堆栈节点数寄存器-1 \*/

}

}

pr->next = NULL; /\* 将末节点的前一个节点置成末节点 \*/

result = p->data;

free(p);

return result;

}

困难9章

按如下函数原型用函数编程解决如下的日期转换问题（要求考虑闰年的问题）：

/\* 函数功能： 对给定的某年某月某日，计算它是这一年的第几天

函数参数： 整型变量year、month、day，分别代表年、月、日

函数返回值：这一年的第几天 \*/

int DayofYear(int year, int month, int day);

/\* 函数功能： 对给定的某一年的第几天，计算它是这一年的第几月第几日

函数入口参数：整型变量year，存储年

整型变量yearDay，存储这一年的第几天

函数出口参数：整型指针pMonth，指向存储这一年第几月的整型变量

整型指针pDay，指向存储第几日的整型变量

函数返回值： 无 \*/

void MonthDay(int year, int yearDay, int \*pMonth, int \*pDay);

输出如下菜单，用switch语句实现根据用户输入的选择执行相应的操作。

1. year/month/day -> yearDay

2. yearDay -> year/month/day

3. Exit

Please enter your choice:

输入提示信息：

"Please enter year, month, day:"

"Please enter year, yearDay:"

输入格式：

"%d,%d,%d"

"%d,%d"

输出提示信息和格式：

"yearDay = %d\n"

"month = %d, day = %d\n"

#include <stdio.h>

#include <stdlib.h>

int DayofYear(int year, int month, int day);

void MonthDay(int year, int yearDay, int \*pMonth, int \*pDay);

void Menu(void);

int dayTab[2][13] = {{0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31}, {0, 31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31}};

int main()

{

int year, month, day, yearDay;

char c;

Menu(); /\* 调用Menu函数显示一个固定式菜单 \*/

c = getchar(); /\* 输入选择 \*/

switch (c) /\* 判断选择的是何种操作 \*/

{

case '1':

printf("Please enter year, month, day:");

scanf("%d,%d,%d", &year, &month, &day);

yearDay = DayofYear(year, month, day);

printf("yearDay = %d\n", yearDay);

break;

case '2':

printf("Please enter year, yearDay:");

scanf("%d,%d", &year, &yearDay);

MonthDay(year, yearDay, &month, &day);

printf("month = %d,day = %d\n", month, day);

break;

case '3':

exit(0); /\* 退出程序的运行 \*/

default:

printf("Input error!");

}

return 0;

}

/\* 函数功能：对给定的某年某月某日，计算并返回它是这一年的第几天 \*/

int DayofYear(int year, int month, int day)

{

int i, leap;

/\* 若year为闰年，即leap值为1，则用第1行元素dayTab[1][i]计算；

否则leap值为0，用第0行dayTab[0][i]计算 \*/

leap = ((year % 4 == 0) && (year % 100 != 0)) || (year % 400 == 0);

for (i = 1; i < month; i++)

{

day = day + dayTab[leap][i];

}

return day; /\* 返回计算出的day的值 \*/

}

/\* 函数功能：对给定的某一年的第几天，计算它是这一年的第几月第几日 \*/

void MonthDay(int year, int yearDay, int \*pMonth, int \*pDay)

{

int i, leap;

leap = ((year % 4 == 0) && (year % 100 != 0)) || (year % 400 == 0);

for (i = 1; yearDay > dayTab[leap][i]; i++)

{

yearDay = yearDay - dayTab[leap][i];

}

\*pMonth = i; /\* 将计算出的月份值赋值给pMonth所指向的变量 \*/

\*pDay = yearDay; /\* 将计算出的日号赋值给pDay所指向的变量 \*/

}

/\* 函数功能：显示菜单 \*/

void Menu(void)

{

printf("1. year/month/day -> yearDay\n");

printf("2. yearDay -> year/month/day\n");

printf("3. Exit\n");

printf("Please enter your choice:");

}

从键盘输入n个整数，用函数编程实现最大值和最小值的交换，并打印结果。

\*\*输入格式要求："%d"

提示信息："Input n(n<=10):" "Input %d numbers:"

\*\*输出结果格式要求："%5d"

提示信息："Exchange results:"

#include <stdio.h>

int main(void)

{

int num[10];

int n, i, tmp;

int max, min;

printf("Input n(n<=10):");

scanf("%d", &n);

printf("Input %d numbers:", n);

for ( i = 0; i < n; i++)

{

scanf("%d", &num[i]);

}

max = 0;

min = 0;

for ( i = 1; i < n; i++)

{

if (num[i] > num[max])

{

max = i;

}

else if (num[i] < num[min])

{

min = i;

}

}

tmp = num[min];

num[min] = num[max];

num[max] = tmp;

printf("Exchange results:");

for ( i = 0; i < n; i++)

{

printf("%5d", num[i]);

}

return 0;

}

困难8章

某班期末考试科目为数学（MT）、英语（EN）和物理（PH），有最多不超过40人参加考试。请编程计算：（1）每个学生的总分和平均分；（2）每门课程的总分和平均分。

\*\*输入格式要求："%d" "%ld"(学号) 提示信息："Input the total number of the students(n<40):" "Input student’s ID and score as: MT EN PH:\n"

\*\*输出格式要求："Counting Result:\n" "Student’s ID\t MT \t EN \t PH \t SUM \t AVER\n" "%12ld\t"(打印学号) "%4d\t"（打印每门课成绩）"%4d\t%5.1f\n"（打印总分和平均分） "SumofCourse \t" "%4d\t"(打印每门课的总分) "\nAverofCourse\t" "%4.1f\t"(每门课的平均分)

程序的运行示例如下：

Input the total number of the students(n<=40):4

Input student’s ID and score as: MT EN PH:

070310122 97 87 92

070310123 92 91 90

070310124 90 81 82

070310125 73 65 80

Counting Result:

Student’s ID MT EN PH SUM AVER

070310122 97 87 92 276 92.0

070310123 92 91 90 273 91.0

070310124 90 81 82 253 84.3

070310125 73 65 80 218 72.7

SumofCourse 352 324 344

AverofCourse 88.0 81.0 86.0

#include <stdio.h>

#define STUD\_N 40 /\* 最多学生人数 \*/

#define COURSE\_N 3 /\* 考试科目数 \*/

int ReadScore(int score[][COURSE\_N], long num[]);

void AverforStud(int score[][COURSE\_N], int sum[], float aver[], int n);

void AverforCourse(int score[][COURSE\_N], int sum[], float aver[], int n);

void Print(int score[][COURSE\_N], long num[], int sumS[], float averS[],

int sumC[], float averC[], int n);

int main()

{

int score[STUD\_N][COURSE\_N], sumS[STUD\_N], sumC[STUD\_N], n;

long num[STUD\_N];

float averS[STUD\_N], averC[STUD\_N];

n = ReadScore(score, num); /\* 读入学生成绩 \*/

AverforStud(score, sumS, averS, n); /\* 计算每个学生的总分平均分 \*/

AverforCourse(score, sumC, averC, n); /\* 计算每门课程的总分平均分 \*/

Print(score, num, sumS, averS, sumC, averC, n); /\* 输出学生成绩 \*/

return 0;

}

/\* 函数功能：输入学生的学号及其三门课的成绩，当输入负值时，结束输入，返回学生人数 \*/

int ReadScore(int score[][COURSE\_N], long num[])

{

int i, j, n;

printf("Input the total number of the students(n<40):");

scanf("%d", &n); /\* 输入参加考试的学生人数 \*/

printf("Input student’s ID and score as: MT EN PH:\n");

for (i = 0; i < n; i++) /\* 对所有学生进行循环 \*/

{

scanf("%ld", &num[i]); /\* 以长整型格式输入每个学生的学号 \*/

for (j = 0; j < COURSE\_N; j++) /\* 对所有课程进行循环 \*/

{

scanf("%d", &score[i][j]); /\* 输入每个学生的各门课成绩 \*/

}

}

return i; /\* 返回学生人数 \*/

}

/\* 函数功能： 计算每个学生的总分和平均分 \*/

void AverforStud(int score[][COURSE\_N], int sum[], float aver[], int n)

{

int i, j;

for (i = 0; i < n; i++)

{

sum[i] = 0;

for (j = 0; j < COURSE\_N; j++) /\* 对所有课程进行循环 \*/

{

sum[i] = sum[i] + score[i][j]; /\* 计算第i个学生的总分 \*/

}

aver[i] = (float)sum[i] / COURSE\_N;/\* 计算第i个学生的平均分 \*/

}

}

/\* 函数功能：计算每门课程的总分和平均分 \*/

void AverforCourse(int score[][COURSE\_N], int sum[], float aver[], int n)

{

int i, j;

for (j = 0; j < COURSE\_N; j++)

{

sum[j] = 0;

for (i = 0; i < n; i++) /\* 对所有学生进行循环 \*/

{

sum[j] = sum[j] + score[i][j]; /\* 计算第j门课程的总分 \*/

}

aver[j] = (float)sum[j] / n; /\* 计算第j门课程的平均分 \*/

}

}

/\*函数功能：打印每个学生的学号、各门课成绩、总分和平均分，以及每门课的总分和平均分\*/

void Print(int score[][COURSE\_N], long num[], int sumS[], float averS[],

int sumC[], float averC[], int n)

{

int i, j;

printf("Counting Result:\n");

printf("Student’s ID\t MT \t EN \t PH \t SUM \t AVER\n");

for (i = 0; i < n; i++)

{

printf("%12ld\t", num[i]); /\* 以长整型格式打印学生的学号 \*/

for (j = 0; j < COURSE\_N; j++)

{

printf("%4d\t", score[i][j]); /\* 打印学生的每门课成绩 \*/

}

printf("%4d\t%5.1f\n", sumS[i], averS[i]);/\*打印学生的总分平均分\*/

}

printf("SumofCourse \t");

for (j = 0; j < COURSE\_N; j++) /\* 打印每门课的总分 \*/

{

printf("%4d\t", sumC[j]);

}

printf("\nAverofCourse\t");

for (j = 0; j < COURSE\_N; j++) /\* 打印每门课的平均分 \*/

{

printf("%4.1f\t", averC[j]);

}

printf("\n");

}

困难函数

梅森尼数

形如2^i-1的素数，称为梅森尼数。编程计算并输出指数i在[2,n]中的所有梅森尼数，并统计这些梅森尼数的个数，其中n的值由键盘输入，并且n的值不能大于50。其中，2^i表示2的i次方，请不要使用pow(2,i)编程计算，应采用循环累乘求积的方式计算2^i。

提示：当i超过30以后，2^i-1的值会很大，不能用long型变量来存储，必须使用double类型来存储。对于double类型变量x（不是整型）不能执行求余运算，即不能用x%i == 0来判断x是否能被i整除，可以使用x/i == (long long)(x/i)来判断x是否能被i整除。

程序运行示例

Input n:

50↙

2^2-1=3

2^3-1=7

2^5-1=31

2^7-1=127

2^13-1=8191

2^17-1=131071

2^19-1=524287

2^31-1=2147483647

count=8

输入提示信息："Input n:\n"

输入格式: "%d"

输出格式： "2^%d-1=%.0lf\n"

"count=%d\n"

#include<stdio.h>

#include<math.h>

int IsPrime(double x);

int main()

{

double t, m;

int count = 0, i, n;

printf("Input n:\n");

scanf("%d", &n);

t = 2;

for (i = 2; i <= n; i++)

{

t = t \* 2;

m = t - 1;

if (IsPrime(m))

{

count++;

printf("2^%d-1=%.0lf\n", i, m);

}

}

printf("count=%d\n", count);

return 0;

}

// 函数功能：判断x是否是素数，若函数返回0，则表示不是素数，若返回1，则代表是素数

int IsPrime(double x)

{

int i, flag = 1;

int squareRoot = (int)sqrt(x);

if (x <= 1) flag = 0; // 负数、0和1都不是素数

for (i = 2; i <= squareRoot && flag; i++)

{

if (x / i == (long long)(x / i)) flag = 0; // 若能被整除，则不是素数

}

return flag;

}

困难简单指针变量

利用指针能直接操作内存的特点来编程输出字符串中每个字符在内存中的存储编码（字符串中可以包含英文数字和汉字）。

程序的运行示例1如下：

请输入一个字符串，长度小于等于100：abcABC012中国人

该字符串的内存编码为： 61 62 63 41 42 43 30 31 32 d6 d0 b9 fa c8 cb

程序的运行示例2如下：

请输入一个字符串，长度小于等于100：12345上山打老虎

该字符串的内存编码为： 31 32 33 34 35 c9 cf c9 bd b4 f2 c0 cf bb a2

提示：

输出格式用"%x "

#include <stdio.h>

#include <string.h>

int main(int argc, char \*argv[])

{

char str[101] = {'\0'};

char \*p1, \*p2;

printf("请输入一个字符串，长度小于等于100：");

scanf("%s", str);

p1 = (char \*)(str + strlen(str));

p2 = (char \*)str;

printf("该字符串的内存编码为： ");

while (p1 > p2)

{

printf("%x ", (unsigned char)\*p2 & 0xff);

p2++;

}

return 0;

}

字符数组

将一个字符串插入至另一个源字符串的某个位置：

将一个字符串2插入到源字符串1中 第一次出现某字符的位置，并打印出形成的新串。

如果 字符串1中找不到输入的字符， 则显示“Not found!”并结束程序。

注：源字符串长度及待插入字符串长度不超过50

提示信息：

printf("Input source string 1:\n")

printf("Input inserted string 2:\n")

printf("Input a letter to locate the index:\n")

输出信息格式：

printf("The new string is:%s")

printf(“Not found!”)

测试样例1：

输入信息：

Input source string 1:

abcdecfg

Input inserted string 2:

\*-\*-\*-\*

Input the a letter to locate the index:

c

输出结果：

The new string is:ab\*-\*-\*-\*cdecfg

测试样例2：

输入信息：

Input source string 1:

abcdecfg

Input inserted string 2:

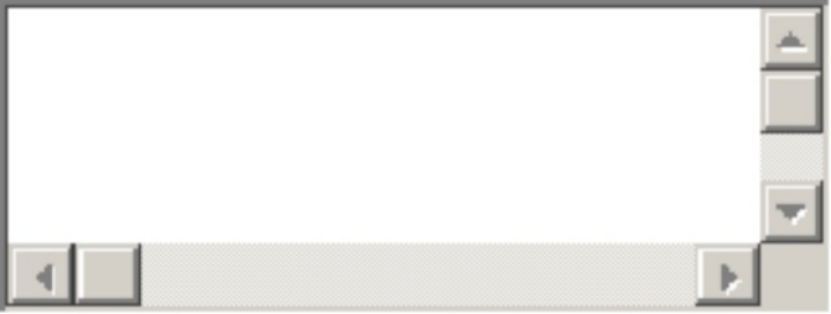
\*\*\*\*

Input the a letter to locate the index:

h

输出结果：

Not found!



1

#include <stdio.h>

#include <string.h> //1

#include <stdlib.h>

/\*将一个字符串插入到另一个字符串中首次出现某字符的指定位置\*/

int main()

{

int i = 0, j = 0, n, len;

char str1[200] = {'\0'};

char str2[100];

char strtem[100];

char ch;

printf("Input source string 1:\n");

gets(str1); //1

printf("Input inserted string 2:\n");

gets(str2); //1

printf("Input a letter to locate the index:\n");

scanf(" %c", &ch); //1

len = strlen(str1); // 1

while (str1[i] != '\0') //1

{

if (str1[i] == ch) //1

{

n = i;

break; //1

}

i++;

}

if (i != len) //1

{

for (i = n; str1[i] != '\0'; j++, i++) //1

{

strtem[j] = str1[i]; //1

str1[i] = '\0'; //1

}

strtem[j] = '\0'; //1

strcat(str1, str2); //1

strcat(str1, strtem); //1

printf("The new string is:%s", str1);

}

else

{

printf("Not found!");

}

return 0;

}

二维数组

找出一个二维数组中的鞍点，即该位置上的元素在该行最大，在该列上最小。也可能没有鞍点。

\*\*输入数据格式：

"\n输入行数："

"%d"

"\n输入列数："

"%d"

"第%d行？\n"

"%d"

\*\*输出格式要求：

"%5d"

"\n第%d行，第%d列的%d是鞍点\n"

"\n矩阵中无鞍点!\n"

程序的运行示例1如下：

输入行数：3

输入列数：3

第0行？

1 2 3

第1行？

4 5 6

第2行？

7 8 9

1 2 3

4 5 6

7 8 9

第0行，第2列的3是鞍点

程序的运行示例2如下：

输入行数：2

输入列数：2

第0行？

1 2

第1行？

4 1

1 2

4 1

矩阵中无鞍点!

#include <stdio.h>

#define N 10

#define M 10

int main()

{

int i, j, k, m, n, flag1, flag2, a[N][M], max, maxi, maxj;

printf("\n输入行数：");

scanf("%d", &n);

printf("\n输入列数：");

scanf("%d", &m);

for (i = 0; i < n; i++)

{

printf("第%d行？\n", i);

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

{

scanf("%d", &a[i][j]);

}

}

for (i = 0; i < n; i++)

{

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

{

printf("%5d", a[i][j]);

}

printf("\n");

}

flag2 = 0;

for (i = 0; i < n; i++)

{

max = a[i][0];

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

{

if (a[i][j] > max)

{

max = a[i][j];

maxj = j;

}

}

for (k = 0, flag1 = 1; k < n && flag1; k++)

{

if (max > a[k][maxj])

{

flag1 = 0;

}

}

if (flag1)

{

printf("\n第%d行，第%d列的%d是鞍点\n", i, maxj, max);

flag2 = 1;

}

}

if (!flag2)

{

printf("\n矩阵中无鞍点!\n");

}

return 0;

}

输入2个学生5门课的成绩，分别用函数求：（1）每个学生的平均分；（2）每门课的平均分；（3）找出最高的分数所对应的学生和课程；（4）求出平均分方差：

(1/n)(∑xi2) - （（∑xi）/ n）2

输入格式和提示信息：

"\n输入学生%4d的5个成绩：\n"

"%f"

输出格式和提示信息：

"\n 序号 课程1 2 3 4 5 平均分\n"

"\n NO%2d"

"%8.2f"

"\n课平均"

"%8.2f"

"\n\n最高分%8.2f是%d号学生的第%d门课\n"

"\n方差 %8.2f"

#include <stdio.h>

#define N 2

#define M 5

float score[N][M];

float a\_stu[N], a\_cor[M];

int main(void)

{

int i, j, r, c;

float h;

float s\_diff(void);

float highest(int \*, int \*);

r = 0;

c = 1;

input\_stu();

avr\_stu();

avr\_cor();

printf("\n 序号 课程1 2 3 4 5 平均分\n");

for (i = 0; i < N; i++)

{

printf("\n NO%2d", i + 1);

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

{

printf("%8.2f", score[i][j]);

}

printf("%8.2f", a\_stu[i]);

}

printf("\n课平均");

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

{

printf("%8.2f", a\_cor[j]);

}

h = highest(&r, &c);

printf("\n\n最高分%8.2f是%d号学生的第%d门课\n", h, r, c);

printf("\n方差 %8.2f", s\_diff());

return 0;

}

input\_stu(void)

{

int i, j;

for (i = 0; i < N; i++)

{

printf("\n输入学生%4d的5个成绩：\n", i + 1);

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

{

scanf("%f", &score[i][j]);

}

}

}

avr\_stu(void)

{

int i, j;

float s;

for (i = 0; i < N; i++)

{

for (j = 0, s = 0; j < M; j++)

{

s += score[i][j];

}

a\_stu[i] = s / 5.0;

}

}

avr\_cor(void)

{

int i, j;

float s;

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

{

s = 0;

for (i = 0; i < N; i++)

{

s += score[i][j];

}

a\_cor[j] = s / (float)N;

}

}

float highest(int \*r, int \*c)

{

float high;

int i, j;

high = score[0][0];

for (i = 0; i < N; i++)

{

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

{

if (score[i][j] > high)

{

high = score[i][j];

\*r = i + 1;

\*c = j + 1;

}

}

}

return high;

}

float s\_diff(void)

{

int i, j;

float sumx, sumxn;

sumx = 0.0;

sumxn = 0.0;

for (i = 0; i < N; i++)

{

sumx += a\_stu[i] \* a\_stu[i];

sumxn += a\_stu[i];

}

return (sumx / N - (sumxn / N) \* (sumxn / N));

}

按如下函数原型，用二维数组作为函数参数，编程计算并输出n×n阶矩阵的转置矩阵。

其中，n由用户从键盘输入。已知n值不超过10。

void Transpose(int a[][N], int n);

void InputMatrix(int a[][N], int n);

void PrintMatrix(int a[][N], int n);

\*\*输入提示信息要求："Input n:\n" "Input %d\*%d matrix:\n"

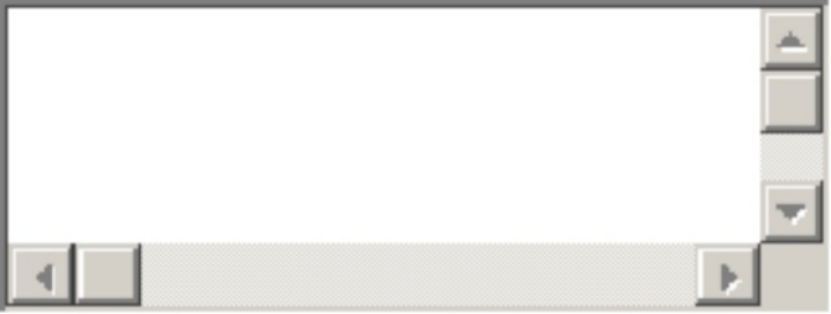
\*\*输入格式："%d"

\*\*输出提示信息："The transposed matrix is:\n"

\*\*输出要求："%d\t"

输出矩阵时，每输出一行就换行

注：不能使用指针、结构体、共用体、文件、goto、枚举类型进行编程。



#include <stdio.h>

#define N 10

void Transpose(int a[][N], int n);

void InputMatrix(int a[][N], int n);

void PrintMatrix(int a[][N], int n);

main()

{

int s[N][N], n;

printf("Input n:\n");

scanf("%d", &n);

InputMatrix(s, n);//1

Transpose(s, n);//1

printf("The transposed matrix is:\n");

PrintMatrix(s, n);//1

}

void Transpose(int a[][N], int n)

{

int i, j, temp;

for (i = 0; i < n; i++) //1

{

for (j = i; j < n; j++)//1

{

temp = a[i][j];//1

a[i][j] = a[j][i];//1

a[j][i] = temp;//1

}

}

}

void InputMatrix(int a[][N], int n) {

int i, j;

printf("Input %d\*%d matrix:\n", n, n);

for (i = 0; i < n; i++) //1

{

for (j = 0; j < n; j++)//1

{

scanf("%d", &a[i][j]);//1

}

}

}

void PrintMatrix(int a[][N], int n)

{

int i, j;

for (i = 0; i < n; i++) //1

{

for (j = 0; j < n; j++) //1

{

printf("%d\t", a[i][j]);//1

}

printf("\n");//1

}

}

矩阵乘法。编写一个C函数实现M行K列矩阵与K行N列的矩阵的乘积。设A为M行K列的矩阵，B为K行N列的矩阵，则C=A×B的积为M行N列的矩阵。

矩阵乘法的规则是：设A[m,k]，B[k,n]，则C[m,n]=A[m,k]×B[k,n]，其中：

C[i,j]=∑kl=1A[i,l]×B[l,j], (i=1,2,…,m j=1,2,…,n)

\*\*输出格式要求："\t%d" "array A=\n" "array B=\n" "array C=\n"

程序运行示例如下：

array A=

1 2 3

4 5 6

array B=

1 1

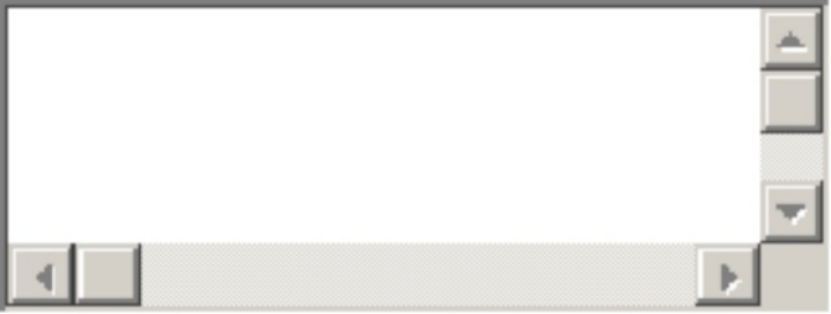
0 2

2 0

array C=

7 5

16 14



1

#include <stdio.h>

main()

{ static int a[2][3]={1,2,3,4,5,6};

static int b[3][2]={1,1,0,2,2,0};

int c[2][2],i,j;

mat (a,b,c,2,2,3);

printf("array A=\n");

printa(a,2,3);

printf("array B=\n");

printa(b,3,2);

printf("array C=\n");

printa(c,2,2);

}

mat(a,b,c,m,n,k)

int m,n,k,a[][3],b[][2],c[][2];

{int i,j,l;

for(i=0;i<m;i++)

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

for(c[i][j]=0,l=0;l<k;l++)

c[i][j]+=a[i][l]\*b[l][j];

}

printa(a,m,n)

int \*a,m,n;

{ int i,j;

for(i=0;i<m;i++,printf("\n"))

for(j=0;j<n;j++,a++)

printf("\t%d",\*a);

}

一维数组

从键盘输入每小时登录网络的用户数到一个有24个整型元素的数组中。写一个程序以如下格式显示一个报告：

时间 登录人数 所占比例

0:00 – 1:00 1 0.3

1:00 – 2:00 2 0.7

…

10:00 - 11:00 27 9.0

11:00 - 12:00 28 9.3

…

23:00 - 24:00 8 2.7

最大登录人数28发生在11：00到12：00之间

最小登录人数 1发生在 0：00到 1：00之间(两个时间段登录人数相同时，取排在前面的时间)

注意：比例后面的%省略，无需输出%，只输出%前面的数即可。

\*\*输入格式要求："%d" 提示信息："请输入在%d:00和%d:00之间登录的用户数："

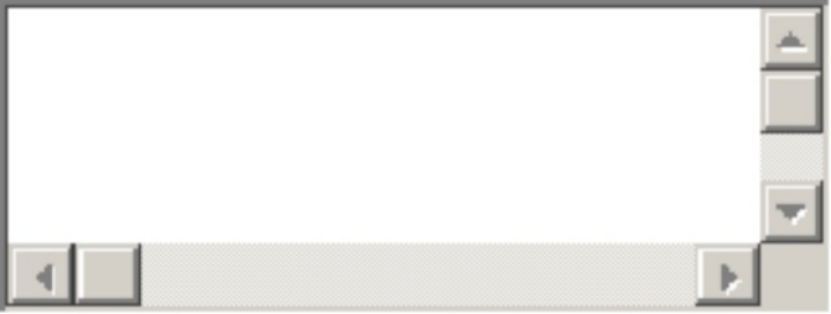
\*\*输出格式要求：

" 时间 登录人数 所占比例\n"

"%2d:00 - %2d:00 %15d %25.1f\n"

"最大登录人数%d发生在%2d：00到%2d：00之间\n"

"最小登录人数%d发生在%2d：00到%2d：00之间\n"



1

#include <stdio.h>

#define N 24

int main()

{

int time\_entered, total\_num\_customers = 0;

int num\_customers[N];

int max\_customers, min\_customers, max, min;

// Enter number of customers for each hour.

for (time\_entered = 0; time\_entered < N; time\_entered++)

{

printf("请输入在%d:00和%d:00之间登录的用户数：",

time\_entered, time\_entered + 1);

scanf("%d", &num\_customers[time\_entered]);

total\_num\_customers += num\_customers[time\_entered] ;

}

// Display report headings.

printf(" 时间 登录人数 所占比例\n");

// Display report lines.

max\_customers = min\_customers = num\_customers[0];

max = min = 0; //记录最大最小的下标

for (time\_entered = 0; time\_entered < N; time\_entered++)

{

printf("%2d:00 - %2d:00 %15d %25.1f\n", time\_entered, time\_entered + 1,

num\_customers[time\_entered], (num\_customers[time\_entered] \* 100.0) / total\_num\_customers);

if (num\_customers[time\_entered] > max\_customers)

{

max\_customers = num\_customers[time\_entered];

max = time\_entered;

}

if (num\_customers[time\_entered] < min\_customers)

{

min\_customers = num\_customers[time\_entered];

min = time\_entered;

}

}

printf("最大登录人数%d发生在%2d：00到%2d：00之间\n", max\_customers, max, max + 1);

printf("最小登录人数%d发生在%2d：00到%2d：00之间\n", min\_customers, min, min + 1);

return 0;

}

满足特异条件的数列。输入m和n（20≥m≥n≥0），求出满足以下方程式的正整数数列i1,i2,…,in，使得i1+i2+…+in=m，且i1≥i2≥…≥in。例如：

当n=4，m=8时，将得到如下5个数列：

5 1 1 1 4 2 1 1 3 3 1 1 3 2 2 1 2 2 2 2

\*\*输入格式要求："%d" 提示信息："Please enter requried terms (<=10):"

" their sum:"

\*\*输出格式要求："There are following possible series:\n" "[%d]:" "%d"

程序运行示例：

Please enter requried terms (<=10): 4 8

their sum:There are following possible series:

[1]:5111

[2]:4211

[3]:3311

[4]:3221

[5]:2222

#define NUM 10

#include <stdio.h>

int i[NUM];

main()

{ int sum, n,total,k,flag,count=0;

printf("Please enter requried terms (<=10):");

scanf("%d",&n);

printf(" their sum:");

scanf("%d",&total);

sum=0;

k=n;

i[n]=1;

printf("There are following possible series:\n");

while(1)

{

if(sum+i[k]<total)

if(k<=1)

{i[1]=total-sum;flag=1;}

else

{sum+=i[k--];

i[k]=i[k+1];

continue;

}

else if(sum+i[k]>total||k!=1)

{sum-=i[++k];

flag=0;

}

else

flag=1;

if(flag)

{

printf("[%d]:",++count);

for(flag=1;flag<=n;++flag)

printf("%d",i[flag]);

printf("\n");

}

if(++k>n)

break;

sum-=i[k];

i[k]++;

}

}

T-primes

time limit per test2 seconds

memory limit per test256 megabytes

inputstandard input

outputstandard output

We know that prime numbers are positive integers that have exactly two

distinct positive divisors. Similarly, we'll call a positive integer t Т-prime,

if t has exactly three distinct positive divisors.

You are given an array of n positive integers. For each of them determine

whether it is Т-prime or not.

Input

The first line contains a single positive integer, n (1≤n≤10^5), showing

how many numbers are in the array. The next line contains n space-separated

integers xi (1≤xi≤10^12).

Please, do not use the %lld specifier to read or write 64bit integers in С++. It is advised to use the cin, cout streams or the %I64d specifier.

Output

Print n lines: the i-th line should contain "YES" (without the quotes), if

number xi is Т-prime, and "NO" (without the quotes), if it isn't.

Sample test(s)

input

3

4 5 6

output

YES

NO

NO

Note

The given test has three numbers. The first number 4 has exactly three di

visors — 1, 2 and 4, thus the answer for this number is "YES". The second

number 5 has two divisors (1 and 5), and the third number 6 has four divi

sors (1, 2, 3, 6), hence the answer for them is "NO".

#include <stdio.h>

#include <string.h>

#include <math.h>

#define eps 1e-6

#define M 1000005

int p[M];

void prime() {

memset(p, 0, sizeof (p));

p[0] = p[1] = 1;

int i, j;

for (i = 2; i < M; i++) {

if (!p[i])

for (j = 2; j \* i < M; j++)

p[i \* j] = 1;

}

}

int main() {

int n;

long long a;

prime();

scanf("%d", &n);

int i;

while (n--) {

scanf("%I64d", &a);

long long t = (long long) (sqrt((double)(a)) + 1 - eps);

if (t \* t == a && !p[t]) {

printf("YES\n");

} else

printf("NO\n");

}

return 0;

}

八皇后问题。在一个8\*8的国际象棋棋盘上，有八个皇后，每个皇后占一格；要求皇后间不会出现相互“攻击”的现象，即不能有两个皇后处在同一行、同一列或同一对角线上。问共有多少种不同的方法。

\*\*输入格式要求：提示信息："The possible configuration of 8 queens are:\n"

\*\*输出格式要求：" [%2d]:" "\n [%2d]:" " %d"

程序运行示例如下：

The possible configuration of 8 queens are:

[ 1]: 1 5 8 6 3 7 2 4 [ 2]: 1 6 8 3 7 4 2 5 [ 3]: 1 7 4 6 8 2 5 3

[ 4]: 1 7 5 8 2 4 6 3 [ 5]: 2 4 6 8 3 1 7 5 [ 6]: 2 5 7 1 3 8 6 4

[ 7]: 2 5 7 4 1 8 6 3 [ 8]: 2 6 8 3 1 4 7 5 [ 9]: 2 6 1 7 4 8 3 5

[10]: 2 7 3 6 8 5 1 4 [11]: 2 7 5 8 1 4 6 3 [12]: 2 8 6 1 3 5 7 4

[13]: 3 5 7 1 4 2 8 6 [14]: 3 5 8 4 1 7 2 6 [15]: 3 5 2 8 1 7 4 6

[16]: 3 5 2 8 6 4 7 1 [17]: 3 6 8 1 4 7 5 2 [18]: 3 6 8 1 5 7 2 4

[19]: 3 6 8 2 4 1 7 5 [20]: 3 6 2 5 8 1 7 4 [21]: 3 6 2 7 1 4 8 5

[22]: 3 6 2 7 5 1 8 4 [23]: 3 6 4 1 8 5 7 2 [24]: 3 6 4 2 8 5 7 1

[25]: 3 7 2 8 5 1 4 6 [26]: 3 7 2 8 6 4 1 5 [27]: 3 8 4 7 1 6 2 5

[28]: 3 1 7 5 8 2 4 6 [29]: 4 6 8 2 7 1 3 5 [30]: 4 6 8 3 1 7 5 2

[31]: 4 6 1 5 2 8 3 7 [32]: 4 7 1 8 5 2 6 3 [33]: 4 7 3 8 2 5 1 6

[34]: 4 7 5 2 6 1 3 8 [35]: 4 7 5 3 1 6 8 2 [36]: 4 8 1 3 6 2 7 5

[37]: 4 8 1 5 7 2 6 3 [38]: 4 8 5 3 1 7 2 6 [39]: 4 1 5 8 2 7 3 6

[40]: 4 1 5 8 6 3 7 2 [41]: 4 2 5 8 6 1 3 7 [42]: 4 2 7 3 6 8 1 5

[43]: 4 2 7 3 6 8 5 1 [44]: 4 2 7 5 1 8 6 3 [45]: 4 2 8 5 7 1 3 6

[46]: 4 2 8 6 1 3 5 7 [47]: 5 7 1 3 8 6 4 2 [48]: 5 7 1 4 2 8 6 3

[49]: 5 7 2 4 8 1 3 6 [50]: 5 7 2 6 3 1 4 8 [51]: 5 7 2 6 3 1 8 4

[52]: 5 7 4 1 3 8 6 2 [53]: 5 8 4 1 3 6 2 7 [54]: 5 8 4 1 7 2 6 3

[55]: 5 1 4 6 8 2 7 3 [56]: 5 1 8 4 2 7 3 6 [57]: 5 1 8 6 3 7 2 4

[58]: 5 2 4 6 8 3 1 7 [59]: 5 2 4 7 3 8 6 1 [60]: 5 2 6 1 7 4 8 3

[61]: 5 2 8 1 4 7 3 6 [62]: 5 3 8 4 7 1 6 2 [63]: 5 3 1 6 8 2 4 7

[64]: 5 3 1 7 2 8 6 4 [65]: 6 8 2 4 1 7 5 3 [66]: 6 1 5 2 8 3 7 4

[67]: 6 2 7 1 3 5 8 4 [68]: 6 2 7 1 4 8 5 3 [69]: 6 3 5 7 1 4 2 8

[70]: 6 3 5 8 1 4 2 7 [71]: 6 3 7 2 4 8 1 5 [72]: 6 3 7 2 8 5 1 4

[73]: 6 3 7 4 1 8 2 5 [74]: 6 3 1 7 5 8 2 4 [75]: 6 3 1 8 4 2 7 5

[76]: 6 3 1 8 5 2 4 7 [77]: 6 4 7 1 3 5 2 8 [78]: 6 4 7 1 8 2 5 3

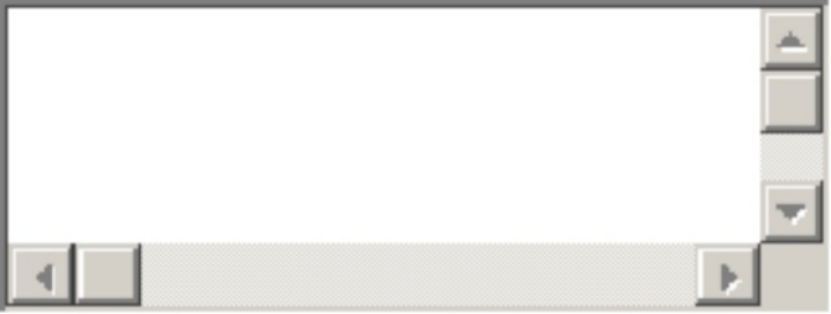
[79]: 6 4 1 5 8 2 7 3 [80]: 6 4 2 8 5 7 1 3 [81]: 7 1 3 8 6 4 2 5

[82]: 7 2 4 1 8 5 3 6 [83]: 7 2 6 3 1 4 8 5 [84]: 7 3 8 2 5 1 6 4

[85]: 7 3 1 6 8 5 2 4 [86]: 7 4 2 5 8 1 3 6 [87]: 7 4 2 8 6 1 3 5

[88]: 7 5 3 1 6 8 2 4 [89]: 8 2 4 1 7 5 3 6 [90]: 8 2 5 3 1 7 4 6

[91]: 8 3 1 6 2 5 7 4 [92]: 8 4 1 3 6 2 7 5



1

#include<stdio.h>

#define NUM 8

int a[NUM+1];

main()

{

int number,i,k,flag,not\_finish=1,count=0;

i=1;

a[1]=1;

printf("The possible configuration of 8 queens are:\n");

while(not\_finish)

{

while(not\_finish&&i<=NUM)

{

for(flag=1,k=1;flag&&k<i;k++)

if(a[k]==a[i]) flag=0;

for(k=1;flag&&k<i;k++)

if((a[i]==a[k]-(k-i))||(a[i]==a[k]+(k-i)))

flag=0;

if(!flag)

{

if(a[i]==a[i-1])

{

i--;

if(i>1&&a[i]==NUM)

a[i]=1;

else if(i==1&&a[i]==NUM)

not\_finish=0;

else a[i]++;

}

else

if(a[i]==NUM)a[i]=1;

else a[i]++;

}

else if(++i<=NUM)

{ if(a[i-1]==NUM)

a[i]=1;

else

a[i]=a[i-1]+1;

}

}

if(not\_finish)

{

++count;

printf((count-1)%3?" [%2d]:" : "\n [%2d]:",count);

for(k=1;k<=NUM;k++)

printf(" %d",a[k]);

if(a[NUM-1]<NUM)

a[NUM-1]++;

else a[NUM-1]=1;

i=NUM-1;

}

}

}

动态数据结构

反向输出一个链表（共5个整型元素）。

程序运行示例如下：

please input 5 data==>

1 2 3 4 5

The value is ==>5

The value is ==>4

The value is ==>3

The value is ==>2

The value is ==>1

#include "stdlib.h"

#include "stdio.h"

struct list

{

int data;

struct list \*next;

} ;

typedef struct list node;

typedef node \*link;

int main()

{

link ptr, head, tail;

int num, i;

tail = (link)malloc(sizeof(node));

tail->next = NULL;

ptr = tail;

printf("\nplease input 5 data==>\n");

for (i = 0; i <= 4; i++)

{

scanf("%d", &num);

ptr->data = num;

head = (link)malloc(sizeof(node));

head->next = ptr;

ptr = head;

}

ptr = ptr->next;

while (ptr != NULL)

{

printf("The value is ==>%d\n", ptr->data);

ptr = ptr->next;

}

}

将一个链表中元素值为x的结点删除。（链表数据域为整数，初始长为6个元素）

程序运行示例如下：

输入数组6个元素的值。

11 22 33 44 55 66

此链表各个结点的数据域为：11 22 33 44 55 66

输入要删除的数据x: 33

删除后链表各个结点的数据域为：11 22 44 55 66

#include <stdio.h>

#include <stdlib.h>

#define N 6

struct LNode

{

int data;

struct LNode \*next;

} ;

struct LNode\* create\_rear(int a[], int n);

void output(struct LNode \*h) ;

struct LNode\* delete\_node(struct LNode\* h, int x);

int main(int argc, char \*argv[])

{

int a[N], i, x;

struct LNode\* head;

printf("输入数组%d个元素的值。\n", N);

for (i = 0; i < N; i++)

scanf("%d", &a[i]);

/\*创建链表head,其结点的值依次为数组a元素的值\*/

head = create\_rear(a, N);

/\*删除前输出链表head\*/

printf("此链表各个结点的数据域为：");

output(head);

printf("输入要删除的数据x: ");

scanf("%d", &x);

head = delete\_node(head, x); /\*调用删除函数\*/

printf("删除后链表各个结点的数据域为：");

output(head); /\*删除后输出链表head\*/

return 0;

}

struct LNode\* create\_rear(int a[], int n)

{

/\*新建一个链表h，每个结点依次插入到链尾，将链表的头指针返回 \*/

struct LNode \*h = NULL;

struct LNode \*s, \*r; /\*用s指向要插入结点，r指向链表的尾结点\*/

int i;

for (i = 0; i < n; i++)

{

s = (struct LNode \*) malloc(sizeof(struct LNode));

s->data = a[i];

s->next = NULL;

if (h == NULL)

h = s; /\*如果链表为空，则头指针h指向s \*/

else

r->next = s; /\*否则将s链接到尾结点r之后 \*/

r = s; /\*将r指向尾结点 \*/

}

return h; /\*返回链表的头指针\*/

}

void output(struct LNode \*h)

{

/\*将链表h的各个结点的数据域依次输出，即遍历该链表\*/

struct LNode \*p = h;/\*从第一个结点开始，用p依次指向各个结点\*/

while (p)

{

/\*只要p是一个非空结点，则输出其数据域，然后将p后移\*/

printf("%d ", p->data);

p = p->next; //将p后移

}

printf("\n");

}

struct LNode\* delete\_node(struct LNode\* h, int x)

{

/\*将链表h中值为x的结点第一个结点删除，并返回头指针。\*/

struct LNode \*pre, \*p;/\*pre所指结点为p所指结点的前驱\*/

p = h;

while (p && x != p->data)

{

/\*如果p不空，且x不等于p所指结点的数据域,p后移，pre为p的前驱\*/

pre = p;

p = p->next;

}

if (p)

{

/\*在链表中找到了要删除的结点p，即p->data为x\*/

if (p == h)

{

/\*p为链首结点，由于p没有前驱，删除后p的后继结点成为链首，需修改头指针\*/

h = p->next;

}

else

{

/\*删除的p非链首结点，则p有前驱pre，删除时需将pre后面链接到p的后继结点\*/

pre->next = p->next;

}

}

return h;

}

超长正整数的加法

请设计一个算法完成两个超长正整数的加法。

\*\*输出格式要求：" s1=" " s2=" "s1+s2="

程序运行示例如下：

3488934387589

374849389

s1=3488934387589

s2=374849389

s1+s2=3489309236978

#include <stdio.h>

#define HUNTHOU 10000

typedef struct node

{ int data;

struct node \*next;

} NODE;

NODE \*insert\_after (u,num)

NODE \*u;

int num;

{ NODE \*v;

v=(NODE \*)malloc(sizeof(NODE));

v->data = num;

u->next = v;

return(v);

}

NODE \* addint(p,q)

NODE \*p,\*q;

{ NODE \*pp,\*qq,\*r,\*s,\*t;

int total,number,carry;

pp=p->next; qq=q->next;

s=(NODE \*)malloc(sizeof(NODE));

s->data=-1;

t=s; carry=0;

while(pp->data!=-1 && qq->data!=-1)

{ total =pp->data + qq->data + carry;

number = total % HUNTHOU;

carry = total / HUNTHOU;

t=insert\_after(t,number);

pp=pp->next;

qq=qq->next;

}

r=(pp->data!=-1)? pp : qq ;

while (r->data!=-1)

{

total = r->data+carry;

number = total % HUNTHOU;

carry = total / HUNTHOU;

t = insert\_after(t,number);

r = r->next;

}

if (carry) t=insert\_after(t,1);

t->next=s;

return(s);

}

NODE \*inputint(void)

{ NODE \*s, \*ps, \*qs;

struct number {int num;

struct number \*np;

} \*p,\*q;

int i,j,k;

long sum;

char c;

p=NULL;

while((c=getchar())!='\n' )

if(c>='0'&&c<='9')

{q=(struct number \*)malloc(sizeof(struct number));

q->num=c-'0';

q->np=p;

p=q;

}

s=(NODE \*)malloc(sizeof(NODE));

s->data=-1;

ps=s;

while(p!=NULL)

{

sum=0;

i=0;

k=1;

while(i<4 && p!=NULL)

{ sum=sum+k\*(p->num);

i++; p=p->np; k=k\*10;

}

qs=(NODE \*)malloc(sizeof(NODE));

qs->data=sum;

ps->next=qs;

ps=qs;

}

ps->next=s;

return(s);

}

printint(s)

NODE \*s;

{ if(s->next->data!=-1 )

{ printint(s->next);

if (s->next->next->data==-1)

printf("%d",s->next->data);

else

{ int i,k=HUNTHOU;

for(i=1;i<=4;i++,k/=10)

putchar('0'+s->next->data%(k)/(k/10));

}

}

}

main()

{ NODE \*s1,\*s2,\*s;

NODE \*inputint(),\*addint(),\*insert\_after();

s1=inputint();

s2=inputint();

printf(" s1=");

printint(s1); putchar('\n');

printf(" s2=");

printint(s2); putchar('\n');

s=addint(s1,s2);

printf("s1+s2=");

printint(s); putchar('\n');

}

将一个链表按逆序排列，即将链头当链尾，链尾当链头。

程序的运行示例如下：

请输入链表（非数表示结束）

结点值：3

结点值：4

结点值：5

结点值：6

结点值：7

结点值：end

原来表：

3 4 5 6 7

反转表：

7 6 5 4 3

#include <stdio.h>

struct line

{

int num;

struct line \*next;

} \*p1, \*p2;

struct line \*creat()

{

int temp;

struct line \*head = NULL;

printf("\n请输入链表（非数表示结束）\n结点值：");

while (scanf("%d", &temp))

{

p1 = (struct line \*)malloc(sizeof(struct line));

(head == NULL) ? (head = p1) : (p2->next = p1);

p1->num = temp;

printf("结点值：");

p2 = p1;

}

p2->next = NULL;

return head;

}

output(struct line \*outhead)

{

for (p1 = outhead; p1 != NULL; printf("%4d", p1->num), p1 = p1->next);

{

}

}

struct line \*turnback(struct line \*head)

{

struct line \*new, \*newhead = NULL;

do

{

p2 = NULL;

p1 = head;

while (p1->next != NULL)

{

p2 = p1;

p1 = p1->next;

}

if (newhead == NULL)

{

newhead = p1;

new = newhead->next = p2;

}

new = new->next = p2;

p2->next = NULL;

}

while (head->next != NULL);

return newhead;

}

int main(void)

{

struct line \*head;

head = creat();

printf("\n原来表：\n");

output(head);

head = turnback(head);

printf("\n\n反转表：\n");

output(head);

return 0;

}

结构体

给定如下定义：

struct date\_rec

{

int day ;

int month ;

int year ;

} ;

struct date\_rec current\_date ;

写一个程序包含如下的函数，完成：

(a) 输入current\_date的值：

void input\_date(struct date\_rec \*current\_date)

(b) 将current\_date增加1天：

void increment\_date(struct date\_rec \*current\_date)

(c) 显示current\_date的值：

void output\_date(struct date\_rec \*current\_date)

考虑每个月的实际天数，同时也考虑闰年的情况。

\*\*输入格式要求："%d%d%d" 提示信息："请输入当前日期（年 月 日）："

\*\*输出格式要求："当前日期：%d年%d月%d日！" （加1天后的新日期）

#include <stdio.h>

struct date\_rec

{

int day;

int month;

int year;

} ;

struct date\_rec current\_date;

int days\_of\_month[][13] = {{0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31}, {0, 31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31}

} ;

int is\_leap(int year)

{

return ((year % 400 == 0) || ((year % 4 == 0) && (year % 100 != 0)));

}

void input\_date(struct date\_rec \*current\_date)

{

printf("请输入当前日期（年 月 日）：");

scanf("%d%d%d", &current\_date->year, &current\_date->month, &current\_date->day);

}

void increment\_date(struct date\_rec \*current\_date)

{

current\_date->day++;

if (current\_date->day > days\_of\_month[is\_leap(current\_date->year)][current\_date->month])

{

current\_date->day = current\_date->day - days\_of\_month[is\_leap(current\_date->year)][current\_date->month]

;

current\_date->month++;

if (current\_date->month > 12)

{

current\_date->year++;

current\_date->month = current\_date->month - 12;

}

}

}

void output\_date(struct date\_rec \*current\_date)

{

printf("当前日期：%d年%d月%d日！", current\_date->year,

current\_date->month, current\_date->day);

}

int main()

{

input\_date(&current\_date);

increment\_date(&current\_date);

output\_date(&current\_date);

return 0;

}

指针变量作函数参数

按如下函数原型用函数编程解决如下的日期转换问题（要求考虑闰年的问题）：

/\* 函数功能： 对给定的某年某月某日，计算它是这一年的第几天

函数参数： 整型变量year、month、day，分别代表年、月、日

函数返回值：这一年的第几天 \*/

int DayofYear(int year, int month, int day);

/\* 函数功能： 对给定的某一年的第几天，计算它是这一年的第几月第几日

函数入口参数：整型变量year，存储年

整型变量yearDay，存储这一年的第几天

函数出口参数：整型指针pMonth，指向存储这一年第几月的整型变量

整型指针pDay，指向存储第几日的整型变量

函数返回值： 无 \*/

void MonthDay(int year, int yearDay, int \*pMonth, int \*pDay);

输出如下菜单，用switch语句实现根据用户输入的选择执行相应的操作。

1. year/month/day -> yearDay

2. yearDay -> year/month/day

3. Exit

Please enter your choice:

输入提示信息：

"Please enter year, month, day:"

"Please enter year, yearDay:"

输入格式：

"%d,%d,%d"

"%d,%d"

输出提示信息和格式：

"yearDay = %d\n"

"month = %d, day = %d\n"

指针变量做函数参数

输入一个整形数，然后按汉语的习惯，将其读出来并输出。如1052，读作：一千零五十二。

输入样例：

1052

输出样例：

一千零五十二

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

const char shuzi[10][4] = {"零", "一", "二", "三", "四", "五", "六", "七", "八", "九"};

const char weishu[5][4] = {"十", "百", "千", "万", "亿"};

void read(char \*str);

int main()

{

char str[11] = {'0'};

scanf("%s", str + 1);

read(str + 1);

return 0;

}

void read(char \*str)

{

if (strlen(str) == 0)

return;

else

{

switch (str[0])

{

case '0':

if (str[1] != '0' && strlen(str) >= 2)

printf("%s", shuzi[0]);

break;

case '1':

if (strlen(str) != 2 && strlen(str) != 6)

printf("%s", shuzi[1]);

else if (\*(str - 1) != '0')

printf("%s", shuzi[1]);

break;

case '2':

printf("%s", shuzi[2]);

break;

case '3':

printf("%s", shuzi[3]);

break;

case '4':

printf("%s", shuzi[4]);

break;

case '5':

printf("%s", shuzi[5]);

break;

case '6':

printf("%s", shuzi[6]);

break;

case '7':

printf("%s", shuzi[7]);

break;

case '8':

printf("%s", shuzi[8]);

break;

case '9':

printf("%s", shuzi[9]);

break;

default:

return;

}

switch (strlen(str))

{

case 9:

printf("%s", weishu[4]);

break;

case 8:

if (str[0] != '0')

printf("%s", weishu[2]);

break;

case 7:

if (str[0] != '0')

printf("%s", weishu[1]);

break;

case 6:

if (str[0] != '0')

printf("%s", weishu[0]);

break;

case 5:

printf("%s", weishu[3]);

break;

case 4:

if (str[0] != '0')

printf("%s", weishu[2]);

break;

case 3:

if (str[0] != '0')

printf("%s", weishu[1]);

break;

case 2:

if (str[0] != '0')

printf("%s", weishu[0]);

break;

case 1:

break;

default:

return;

}

read(++str);

}

}

用指针编程实现3X4的二维数组的元素读入以及求此二维数组的最大值及最大值下标

请用以下函数实现：

void Input(int \*p,int m, int n); /\*数组元素读入函数\*/

int FindMax(int \*p, int m, int n, int \*pRow, int \*pCol); /\*求最大值及下标函数\*/

若存在若干个相同最大元素，则按照第一次出现的最大值的下标输出。

\*\*\*输入提示信息："Please input your data:\n"

\*\*\*输入格式要求：无格式要求

\*\*\*输出格式要求："The maximum is %d, which is in row %d, colum %d\n"

样例：

Please input your data:

\*\*输入样例：

3 5 2 7 1 6 12 11 4 10 8 9

\*\*输出样例：

The maximum is 12, which is in row 1, colum 2

#include <stdio.h>

#define M 3

#define N 4

void Input(int \*p, int m, int n); //1

int FindMax(int \*p, int m, int n, int \*pRow, int \*pCol); //1

main()

{

int score[M][N], maxScore, row, col;

printf("Please input your data:\n");

Input(\*score, M, N); //1

maxScore = FindMax(\*score, M, N, &row, &col); //2

printf("The maximum is %d, which is in row %d, colum %d\n", maxScore, row, col);

}

void Input(int \*p, int m, int n)

{

int i, j;

for (i = 0; i < m; i++) //1

{

for (j = 0; j < n; j++) //1

scanf("%d", &p[i \* n + j]);

}

}

int FindMax(int \*p, int m, int n, int \*pRow, int \*pCol )

{

int i, j, max;

max = p[0]; //1

\*pRow = 0;

\*pCol = 0;

for ( i = 0; i < m; i++ ) //1

{

for ( j = 0; j < n; j++ ) //1

{

if ( p[ i \* n + j ] > max ) //1

{

max = p[ i \* n + j]; //1

\*pRow = i;

\*pCol = j;

}

}

}

return max;

}

输入3个数x,y,z，按从小到大顺序排序后输出。

要求：利用指针方法实现两数互换，函数原型为：void swap(int \*p1,int \*p2);

输入提示：printf("please input 3 number x,y,z");

输出格式：printf("the sorted numbers are:%d,%d,%d\n", );

#include <stdio.h>

void swap(int \*p1,int \*p2); //1

main()

{

int x,y,z;

int \*p1,\*p2,\*p3;

printf("please input 3 number x,y,z");

scanf("%d,%d,%d",&x,&y,&z); //1

p1=&x; //1

p2=&y; //1

p3=&z; //1

if(x>y)swap(p1,p2); //1

if(x>z)swap(p1,p3); //1

if(y>x)swap(p2,p3); //1

printf("the sorted numbers are:%d,%d,%d\n",x,y,z);

}

void swap(int \*p1,int \*p2)

{

int temp;

temp=\*p1; //1

\*p1=\*p2; //1

\*p2=temp; //1

}

字符数组作函数参数

写一个函数，输入一行字符，将此字符串中最长的单词输出。

程序的运行示例如下：

输入一行文本：I am a student.

最长的单词是：student

#include <stdio.h>

int alphabetic(char c)

{

if ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z'))

{

return 1;

}

else

{

return 0;

}

}

/\* 寻找最长单词 \*/

int longest(char string[])

{

int len = 0, i, length = 0, flag = 1, place, point;

for (i = 0; i <= strlen(string); i++)

{

if (alphabetic(string[i]))

{

if (flag)

{

point = i;

flag = 0;

}

else

{

len++;

}

}

else

{

flag = 1;

if (len > length)

{

length = len;

place = point;

len = 0;

}

}

}

return place;

}

int main()

{

int i;

char line[100];

printf("输入一行文本：\n");

gets(line);

printf("\n最长的单词是：");

for (i = longest(line); alphabetic(line[i]); i++)

{

printf("%c", line[i]);

}

printf("\n");

return 0;

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

对输入的字符串进行长度验证，保证输入的字符串的长度在指定的范围内，如果不在指定的范围内，则一直提示用户输入，直到输入合法长度的字符串为止。

程序的示例运行如下：

请输入一个字符串(长度为[3..5]个字符)：a

请输入一个字符串(长度为[3..5]个字符)：ab

请输入一个字符串(长度为[3..5]个字符)：abcdef

请输入一个字符串(长度为[3..5]个字符)：abc

你输入的字符串为：abc

#include <stdio.h>

void getString(char\* str, int min, int max)

{

int len;

printf("请输入一个字符串(长度为[%d..%d]个字符)：\n", min, max);

scanf("%s", str);

len = strlen(str);

while (len < min || len > max)

{

printf("请输入一个字符串(长度为[%d..%d]个字符)：\n", min, max);

scanf("%s", str);

len = strlen(str);

}

}

int main(int argc, char\* argv[])

{

char string[100];

getString(string, 3, 5);

printf("你输入的字符串为：%s\n", string);

return 0;

}

请按给定的函数原型编程实现将字符数组中的字符串的第m个字符开始的n个字符逆序存放。要求在主函数读入字符串，且逆序存放后的字符串也在主函数打印。函数原型：

void inverse(char str[],int m,int n);

输入要求: 输入的一行字符串，应包含字母，数字以及空格字符

友情提示:

在执行输入字符串的函数之前,请用getchar();把输入缓冲区中的换行符读出!!!

\*\*\*\*输入提示信息和格式要求为：

"input m,n:"

"%d,%d"

"input the string:"

\*\*\*\*输出格式为:"the inverse string:%s"

#include <stdio.h>

#define N 100

void inverse(char str[], int m, int n);

main()

{

char str[N];

int m, n;

printf("input m,n:"); //1

scanf("%d,%d", &m, &n);

getchar();

printf("input the string:"); //1

gets(str);

inverse(str, m, n); //2

printf("the inverse string:%s", str); //1

}

void inverse(char str[], int m, int n)

{

int i;

char a[N];

for (i = 0; i < n; i++)

{

a[i] = str[m - 1 + i]; //1

}

for (i = 0; i < n; i++)

{

str[m + i - 1] = a[n - 1 - i]; //1

}

}

写一个函数，将一个字符串中的元音字母复制到另一个字符串，然后输出。

程序的运行示例如下：

提示信息："\n输入字符串："

字符串中的元音字母是eoo

\*\*\*输入数据格式\*\*\*：使用gets()

\*\*\*输出数据格式\*\*\*："\n字符串中的元音字母是%s"

程序运行示例：

输入字符串：Hello,world!↙

Hello,world!

字符串中的元音字母是eoo

#include <stdio.h>

int main()

{

char str[80], c[80];

void cpy();

printf("\n输入字符串：");

gets(str);

printf("%s", str);

cpy(str, c);

printf("\n字符串中的元音字母是%s", c);

return 0;

}

void cpy(char s[], char c[])

{

int i, j;

for (i = 0, j = 0; s[i] != '\0'; i++)

{

if (s[i] == 'a' || s[i] == 'A' || s[i] == 'e' || s[i] == 'E' ||

s[i] == 'i' || s[i] == 'I' || s[i] == 'o' || s[i] == 'O' ||

s[i] == 'u' || s[i] == 'U')

{

c[j] = s[i];

j++;

}

}

c[j] = '\0';

}

编写一个程序，将用户输入的由数字字符和非数字字符组成的字符串中的数字提取出来（例如：输入asd123,34fgh\_566kkk789，则产生的数字分别是123、34、789）。

\*\*输入格式要求：提示信息："Please enter a string:"

\*\*输出格式要求："the result of output:\n" "%10d\n"

程序运行示例如下：

Please enter a string:

abc123def456ghi111bbbccc99go100

the result of output:

123

456

111

99

100

#include <stdio.h>

#include <stdlib.h>

#define N 100

int getline(char \*s, int len);

int main()

{

char line[N], b[20], \*ptr1, \*ptrb;

int n = 0, j, a[50];

printf("Please enter a string:");

getline(line, N);

ptr1 = line;

while (\*ptr1 != '\0')

{

ptrb = b;

for (j = 0; \*ptr1 >= '0' && \*ptr1 <= '9'; ptr1++, ptrb++, j++)

{

\*ptrb = \*ptr1;

}

if (j)

{

\*ptrb = '\0';

\*(a + n) = atoi(b);

n++;

}

if (\*ptr1 != '\0')ptr1++;

}

printf("the result of output:\n");

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

{

printf("%10d\n", \*(a + j));

}

return 0;

}

int getline(char \*s, int len)

{

int c;

char \*p = s;

while (--len > 0 && (c = getchar()) != '\n')

{

\*s++ = c;

}

\*s = '\0';

return s - p;

}

题目：去除字符串中的空格并分解单词。

要求：在main()中调用下述各函数，完成字符串中去除空格并分解单词的功能。

函数原型如下：

函数一：单词内是否包含数字 int IsNumIn(char word[])

函数二：单词内是否包含空格 int IsSpcIn(char word[])

函数三：去掉单词的前后空格，tab键和换行符 Trim(char oldWord[], char newWord[])

函数四：单词内部有空格，分解成多个单词 Seg(char words[], char wArray[][100] ) 假设单词内部只有一个空格，没有两个连续空格的情况发生。

要求利用gets函数来获得用户输入的字符串，不能用scanf来获得用户输入的字符串

参考输入，输出

参考输入

ros2e

参考输出

error

参考输入

︺︺hello︺（ ︺代表一个空格）

参考输出

hello

参考输入

︺︺hello world︺

参考输出

hello

world

#include <stdio.h>

#include <ctype.h>

#define N 100

int IsNumIn(char word[]);

int IsSpcIn(char word[]);

void Trim(char oldWord[], char newWord[]);

void Seg(char words[], char wArray[][100]);

int main()

{

char str1[N], str[N], w\_Arr[20][100];

int i, j;

for (i = 0; i < N; i++)

{

\*(str1 + i) = 0;

\*(str + i) = 0;

}

for (i = 0; i < 20; i++)

{

for (j = 0; j < 100; j++)

w\_Arr[i][j] = 0;

}

gets(str1);

Trim(str1, str);

if (IsNumIn(str))

printf("error");

else

{

if (IsSpcIn(str))

{

Seg(str, w\_Arr);

for (i = 0; w\_Arr[i][0] != '\0'; i++)

{

puts(w\_Arr[i]);

}

}

else

puts(str);

}

return 0;

}

int IsNumIn(char word[])

{

int i;

for (i = 0; \*(word + i) != '\0'; i++)

{

if (isdigit(\*(word + i)))

return 1;

}

return 0;

}

int IsSpcIn(char word[])

{

int i;

for (i = 0; \*(word + i) != '\0'; i++)

{

if (isspace(\*(word + i)))

return 1;

}

return 0;

}

void Trim(char oldWord[], char newWord[])

{

int i = 0, j = 0;

while (isspace(\*(oldWord + i)))

i++;

while (\*(oldWord + i) != '\0')

{

\*(newWord + j) = \*(oldWord + i);

i++;

j++;

}

i--;

j--;

for (; isspace(\*(oldWord + i)); i--, j--)

{

\*(newWord + j) = 0;

}

}

void Seg(char words[], char wArray[][100])

{

int i, j = 0, k = 0;

for (i = 0; i < N; i++)

{

if (isalpha(\*(words + i)))

{

wArray[j][k] = \*(words + i);

k++;

}

else

{

j++;

k = 0;

}

}

}

程序中函数 fun()的功能: 将一个由八进制数字字符组成的字符串转换为与其值相等的十进制整数。规定输入的字符串最多只能包含5位八进制数字字符。

\*\*输入格式要求："%d" 提示信息："输入一个八进制字符串（5位）:" "错误：字符串太长\n\n" "错误:%c 不是一个八进制字符\n\n"

\*\*输出格式要求："输入的八进制字符串为" "\n%s 转换成十进制整数为：%d\n"

程序运行示例如下：

输入一个八进制字符串（5位）:234

输入的八进制字符串为234

234 转换成十进制整数为：156

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

int fun(char\* p)

{

int n;

n = \*p - '0';

p++;

while (\*p != 0)

{

n = n \* 8 + \*p - '0';

p++;

}

return n;

}

main()

{

char s[6];

int i;

int n;

printf("输入一个八进制字符串（5位）:");

gets(s);

if (strlen(s) > 5)

{

printf("错误：字符串太长\n\n");

exit(0);

}

for (i = 0; s[i]; i++)

{

if (s[i] < '0' || s[i] > '7')

{

printf("错误:%c 不是一个八进制字符\n\n", s[i]);

exit(0);

}

}

printf("输入的八进制字符串为");

puts(s);

n = fun(s);

printf("\n%s 转换成十进制整数为：%d\n", s, n);

}

利用一个字符数组作函数参数,实现字符串（最大长度为80个字符 ）的逆序存放。

要求如下：

(1)在子函数Inverse中实现字符串的逆序存放。函数原型为：

void Inverse(char str[]);

(2)在主函数中

从键盘输入字符串(使用gets函数)

然后，调用Inverse函数，

最后，输出逆序后的字符串。

(3)\*\*输入提示信息："Input a string:\n"

\*\*输出提示信息："Inversed results:\n"

\*\*输出格式："%s\n"

#include <stdio.h>

#include <string.h>

#define N 80

void Inverse(char str[]);

main()

{

char a[N];

printf("Input a string:\n");

gets(a);//1

Inverse(a);//2

printf("Inversed results:\n");

puts(a);//1

}

void Inverse(char str[])

{

int len, i, j;

char temp;//1

len = strlen(str);//1

for (i = 0, j = len - 1; i < j; i++, j--)//4

{

temp = str[i];//1

str[i] = str[j];//1

str[j] = temp;//1

}

}

写一函数，输入一个十六进制数，输出相应的十进制数。

程序的运行示例如下：

输入十六进制数：

3a

十进制数58

继续吗？

y

输入十六进制数：

2f

十进制数47

继续吗？

N

#include <stdio.h>

#define MAX 1000

int main()

{

int c, i, flag, flag1;

char t[MAX];

i = 0;

flag = 0;

flag1 = 1;

printf("输入十六进制数：");

while ((c = getchar()) != '\0' && i < MAX && flag1)

{

if (c >= '0' && c <= '9' || c >= 'a' && c <= 'f' || c >= 'A' && c <= 'F')

{

flag = 1;

t[i++] = c;

}

else if (flag)

{

t[i] = '\0';

printf("\n十进制数%d\n", htoi(t));

printf("继续吗？");

c = getchar();

if (c == 'N' || c == 'n')

{

flag1 = 0;

}

else

{

flag = 0;

i = 0;

printf("\n输入十六进制数：");

}

}

}

return 0;

}

htoi(char s[])

{

int i, n;

n = 0;

for (i = 0; s[i] != '\0'; i++)

{

if (s[i] >= '0' && s[i] <= '9')

{

n = n \* 16 + s[i] - '0';

}

if (s[i] >= 'a' && s[i] <= 'f')

{

n = n \* 16 + s[i] - 'a' + 10;

}

if (s[i] >= 'A' && s[i] <= 'F')

{

n = n \* 16 + s[i] - 'A' + 10;

}

}

return n;

}

实现验证输入的必须全部是数字0～9构成的字符串，利用函数checkNumber验证输入的字符串中的每个字符是否全为数字0～9之一，如果是则返回1，否则返回0。然后利用函数getNumberString容错地读取一个全由数字0～9组成的数字字符串。要求编写完整的程序测试运行你所写的函数。

输入语句为："请输入一个全为数字的字符串:\n“

输出语句为："你输入的正确的数字串为：%s\n"

#include <stdio.h>

//检查是否全是数字构成的串

int checkNumber(char\* str)

{

int i;

for (i = 0; i < strlen(str); i++)

{

if (str[i] < '0' || str[i] > '9')

{

return 0;

}

}

return 1;

}

char\* getNumberString()

{

static char string[100];

do

{

printf("请输入一个全为数字的字符串:\n");

scanf("%s", string);

} while (!checkNumber(string));

return string;

}

int main(int argc, char\* argv[])

{

char input[100];

strcpy(input, getNumberString());

printf("你输入的正确的数字串为：%s\n", input);

return 0;

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

自己定义一个函数，用字符数组作为函数参数的类型，

以实现和函数strlen相同的功能，即在主函数中任意输入

一个字符串（长度不超过80个字符），调用该函数计算

输入字符串的实际长度，然后打印计算结果。

#include <stdio.h>

#include <string.h>

int Mystrlen(char str[]);

main()

{

char str[80];

int l;

printf("Enter a string:");

gets(str);

l = Mystrlen(str);

printf("The length of the string is:%d\n", l);

}

int Mystrlen(char str[])

{

int i;

for (i = 0; str[i] != '\0'; i++)

{

}

return i;

}

题目：去除字符串中的空格并分解单词。

要求：在main()中调用下述各函数，完成字符串中去除空格并分解单词的功能。

函数原型如下：

函数一：单词内是否包含数字 int IsNumIn(char word[])

函数二：单词内是否包含空格 int IsSpcIn(char word[])

函数三：去掉单词的前后空格，tab键和换行符 Trim(char oldWord[], char newWord[])

函数四：单词内部有空格，分解成多个单词 Seg(char words[], char wArray[][100] ) 假设单词内部只有一个空格，没有两个连续空格的情况发生。

要求利用gets函数来获得用户输入的字符串，不能用scanf来获得用户输入的字符串

#include <stdio.h>

#include <ctype.h>

#define N 100

int IsNumIn(char word[]);

int IsSpcIn(char word[]);

void Trim(char oldWord[], char newWord[]);

void Seg(char words[], char wArray[][100]);

int main()

{

char str1[N], str[N], w\_Arr[20][100];

int i, j;

for (i = 0; i < N; i++)

{

\*(str1 + i) = 0;

\*(str + i) = 0;

}

for (i = 0; i < 20; i++)

{

for (j = 0; j < 100; j++)

w\_Arr[i][j] = 0;

}

gets(str1);

Trim(str1, str);

if (IsNumIn(str))

printf("error");

else

{

if (IsSpcIn(str))

{

Seg(str, w\_Arr);

for (i = 0; w\_Arr[i][0] != '\0'; i++)

{

puts(w\_Arr[i]);

}

}

else

puts(str);

}

return 0;

}

int IsNumIn(char word[])

{

int i;

for (i = 0; \*(word + i) != '\0'; i++)

{

if (isdigit(\*(word + i)))

return 1;

}

return 0;

}

int IsSpcIn(char word[])

{

int i;

for (i = 0; \*(word + i) != '\0'; i++)

{

if (isspace(\*(word + i)))

return 1;

}

return 0;

}

void Trim(char oldWord[], char newWord[])

{

int i = 0, j = 0;

while (isspace(\*(oldWord + i)))

i++;

while (\*(oldWord + i) != '\0')

{

\*(newWord + j) = \*(oldWord + i);

i++;

j++;

}

i--;

j--;

for (; isspace(\*(oldWord + i)); i--, j--)

{

\*(newWord + j) = 0;

}

}

void Seg(char words[], char wArray[][100])

{

int i, j = 0, k = 0;

for (i = 0; i < N; i++)

{

if (isalpha(\*(words + i)))

{

wArray[j][k] = \*(words + i);

k++;

}

else

{

j++;

k = 0;

}

}

}

对输入的8个字符串(每个字符串长度不超过20)按照字典顺序进行排序并输出。不限定排序方法，不能使用goto语句。

输入输出格式：

输入：%s

输出：%s\n

输入：多个字符串用空格分隔

输出：排序好的字符串，每行一个字符串，末尾有空行。

#include<stdio.h>

#include<string.h>

#define MAX 8

#define CMAX 10

int main()

{

char a[MAX][CMAX], term[CMAX]; //1

int i, j;

for (i = 0; i < 8; i++)

{

scanf("%s", a[i]); //1

}

for (i = 0; i < 8 - 1; i++) //1

{

for (j = 1; j < 8 - i; j++) //1

{

if (strcmp(a[j], a[j - 1]) < 0) //1

{

strcpy(term, a[j]); //1

strcpy(a[j], a[j - 1]); //1

strcpy(a[j - 1], term); //1

}

}

}

for (i = 0; i < 8; i++)

{

printf("%s\n", a[i]);

}

return 0;

}

一维数组做函数参数

黑白子交换。有三个白子和三个黑子如下图布置：(-表示一个空格)

○ ○ ○ - ● ● ●

游戏的目的是用多少的步数将上图中白子和黑子的位置进行交换：

● ● ● - ○ ○ ○

游戏的规则是：一次只能移动一个棋子；棋子可以向空格中移动，也可以跳过一个对方的格子进入空格，但不能向后跳，也不能跳过两个子。请用计算机实现上述游戏。

程序运行示例如下：

No. 0:-----------

|\*|\*|\*| |@|@|@|

-----------

No. 1:-----------

|\*|\*| |\*|@|@|@|

-----------

No. 2:-----------

|\*|\*|@|\*| |@|@|

-----------

No. 3:-----------

|\*|\*|@|\*|@| |@|

-----------

No. 4:-----------

|\*|\*|@| |@|\*|@|

-----------

No. 5:-----------

|\*| |@|\*|@|\*|@|

-----------

No. 6:-----------

| |\*|@|\*|@|\*|@|

-----------

No. 7:-----------

|@|\*| |\*|@|\*|@|

-----------

No. 8:-----------

|@|\*|@|\*| |\*|@|

-----------

No. 9:-----------

|@|\*|@|\*|@|\*| |

-----------

No.10:-----------

|@|\*|@|\*|@| |\*|

-----------

No.11:-----------

|@|\*|@| |@|\*|\*|

-----------

No.12:-----------

|@| |@|\*|@|\*|\*|

-----------

No.13:-----------

|@|@| |\*|@|\*|\*|

-----------

No.14:-----------

|@|@|@|\*| |\*|\*|

-----------

No.15:-----------

|@|@|@| |\*|\*|\*|

-----------

int number;

print(a)

int a[];

{

int i;

printf("No.%2d:-----------\n",number++);

printf(" ");

for(i=0;i<=6;i++)

printf("|%c",a[i]==1?'\*':(a[i]==2?'@':' '));

printf("|\n -----------\n");

}

main()

{

int t[7]={1,1,1,0,2,2,2};

int i,flag;

print(t);

while(t[0]+t[1]+t[2]!=6||t[4]+t[5]+t[6]!=3)

{

flag=1;

for(i=0;flag&&i<5;i++)

if(t[i]==1&&t[i+1]==2&&t[i+2]==0)

{change(&t[i],&t[i+2]);print(t);flag=0;}

for(i=0;flag&&i<5;i++)

if(t[i]==0&&t[i+1]==1&&t[i+2]==2)

{change(&t[i],&t[i+2]);print(t);flag=0;}

for(i=0;flag&&i<6;i++)

if(t[i]==1&&t[i+1]==0&&(i==0||t[i-1]!=t[i+2]))

{change(&t[i],&t[i+1]);print(t);flag=0;}

for(i=0;flag&&i<6;i++)

if(t[i]==0&&t[i+1]==2&&(i==5||t[i-1]!=t[i+2]))

{change(&t[i],&t[i+1]);print(t);flag=0;}

}

}

change(n,m)

int \*n,\*m;

{

int term;

term=\*n;\*n=\*m;\*m=term;

}

递归函数

\*汉诺塔问题是一个著名的问题，初始模型如图所示。其来源据说是在约19世纪末欧洲的商店中出售一种智力玩具，在一块铜板上有三根杆，最左边的杆自上而下、由小到大顺序串着64个圆盘构成的塔，游戏的目的是将最左边A杆上的圆盘，借助最右边的C杆，全部移动到中间的B杆上，条件是一次仅能移动一个盘，且不允许大盘放在小盘的上面。

\*\*输入格式要求："%d" 提示信息："Please enter the number of discs:"

\*\*输出格式要求："\tTotal:%d\n" "%2d-(%2d):%c==>%c\n"

程序运行示例如下：

Please enter the number of discs:

Please enter the number of discs: 5

1-( 1):a==>b

2-( 2):a==>c

3-( 1):b==>c

4-( 3):a==>b

5-( 1):c==>a

6-( 2):c==>b

7-( 1):a==>b

8-( 4):a==>c

9-( 1):b==>c

10-( 2):b==>a

11-( 1):c==>a

12-( 3):b==>c

13-( 1):a==>b

14-( 2):a==>c

15-( 1):b==>c

16-( 5):a==>b

17-( 1):c==>a

18-( 2):c==>b

19-( 1):a==>b

20-( 3):c==>a

21-( 1):b==>c

22-( 2):b==>a

23-( 1):c==>a

24-( 4):c==>b

25-( 1):a==>b

26-( 2):a==>c

27-( 1):b==>c

28-( 3):a==>b

29-( 1):c==>a

30-( 2):c==>b

31-( 1):a==>b

Total:31

#include <stdio.h>

void movedisc(unsigned n, char fromneedle, char toneedle, char usingneedle);

int i = 0;

int main()

{

unsigned n;

printf("Please enter the number of discs:");

scanf("%d", &n);

movedisc(n, 'a', 'b', 'c');

printf("\tTotal:%d\n", i);

return 0;

}

void movedisc(unsigned n, char fromneedle, char toneedle, char usingneedle)

{

if (n == 1)

printf("%2d-(%2d):%c==>%c\n", ++i, n, fromneedle, toneedle);

else

{

movedisc(n - 1, fromneedle, usingneedle, toneedle);

printf("%2d-(%2d):%c==>%c\n", ++i, n, fromneedle, toneedle);

movedisc(n - 1, usingneedle, toneedle, fromneedle);

}

}

给下面程序改错。程序功能是输入10个数，按从小到大顺序排序。

#include <stdio.h>

#define SIZE 10;

main( )

{ int i, j, t, a[SIZE];

printf("input 10 numbers: \n");

for(i = 1;i <= SIZE;i++)

scanf("%d", a[i]);

printf("\n");

for (i = 0; i< SIZE; i++)

for (j = SIZE - 1; j >= i + 1; j--)

if (a[j] > a[j-1])

{

t=a[j];

a[j-1]=a[j];

a[j]=t;

}

for (i = 0; i < SIZE; i++)

printf("%d\n", a[i]);

}

注意：

（1）请将修改正确后的完整源程序拷贝粘贴到答题区内。

（2）对于没有错误的语句，请不要修改，修改原本正确的语句也要扣分。

（3）当且仅当错误全部改正，且程序运行结果调试正确，才给加5分。

（4）改错时不能改变程序原有的意图，也不要改变代码的输入输出格式。

#include <stdio.h>

#define SIZE 10

void main( )

{

int i, j, t, a[SIZE];

printf("input 10 numbers: \n");

for (i = 0; i < SIZE; i++)

scanf("%d", &a[i]);

printf("\n");

for (i = 0; i< SIZE-1; i++)

for (j = SIZE - 1; j >= i + 1; j--)

if (a[j] < a[j-1])

{

t=a[j];

a[j]=a[j-1];

a[j-1]=t;

}

for (i = 0; i < SIZE; i++)

printf("%d\n", a[i]);

}

创建动态链表并打印链表内容。要求通过键盘读入学生的学号和分数，当输入的学号为0时表示链表创建结束，并将读入的学生学号和成绩打印输出。下面程序中存在比较隐蔽的错误，请通过分析和调试程序，发现并改正程序中的错误。改错时不能改变程序原有的意图。

注意：将修改后的完整的源程序写在答题区内。

#include <stdio.h>

#define SIZE sizeof(struct node)

struct node

{

long num;

float score;

struct node \*next;

}

int main()

{

int n = 0;

struct node \*head = NULL, \*p1, \*p2;

p1 = p2 = (struct node\*)malloc(SIZE);

printf("Input %d node data:\n", n + 1);

scanf("%d%f", &p1->num, &p1->score);

while (p1->num != 0)

{

if (++n == 1)

head = p1;

else

p2->next = p1;

p1 = p2;

p1 = (struct node\*)malloc(SIZE);

printf("Input %d node data:\n", n + 1);

scanf("%d%f", &p1->num, &p1->score);

}

free(p1);

printf("Prit list:\n");

p2->next = NULL;

if (head != NULL)

{

p2=head;

while (p1 != NULL)

{

printf("num:%d\tscore:%.2f\n", p1->num, p1->score);

p1 = p1->next;

}

}

return 0;

}

#include <stdio.h>

#include <stdlib.h>

#define SIZE sizeof(struct node)

struct node

{

long num;

float score;

struct node \*next;

} ;

int main()

{

int n = 0;

struct node \*head = NULL, \*p1, \*p2;

p1 = p2 = (struct node\*)malloc(SIZE);

printf("Input %d node data:\n", n + 1);

scanf("%d%f", &p1->num, &p1->score);

while (p1->num != 0)

{

if (++n == 1)

head = p1;

else

p2->next = p1;

p2 = p1;

p1 = (struct node\*)malloc(SIZE);

printf("Input %d node data:\n", n + 1);

scanf("%d%f", &p1->num, &p1->score);

}

free(p1);

printf("Prit list:\n");

p2->next = NULL;

if (head != NULL)

{

p1 = head;

while (p1 != NULL)

{

printf("num:%d\tscore:%.2f\n", p1->num, p1->score);

p1 = p1->next;

}

}

return 0;

}

班级共有 m个人，该班C语言的成绩存放在score（score为整数）数组中，计算该班成绩的平均分，并将小于平均分的成绩存储在一个数组中，并打印该数组的值。

要求：

1. 请编写函数fun, 它的功能是：计算平均分，并将低于平均分的成绩和相应的数组下标分别存在不同的数组中(打印语句放在主函数中)，声明如下：

int fun(int score[], int m, int below\_score[], int below\_index[]);

2. 请编写函数ReadScore，读入成绩，返回输入的有效人数，声明如下：

int ReadScore(int score[]);

3. 需要对数组越界做判断，如：在输入时，直接输入-1的情况，此时显示“there are no valid scores”，并终止程序。

4. 班机最多有40人，用宏定义数组的所含最多的元素数量。

输入：每一行输入一个人的成绩，直到输入值为负数时，结束成绩的输入，并将此时拥有的成绩数量，作为班机人数，如：

45

67

98

-1

输出：打印班机的总人数，低于平均分的，低于平均分的成绩及该成绩在输入时的序号，从1开始计数，如：

the number of the class:3

the number under the average score: 2

the 1th score is: 45

the 2th score is: 67

#include <stdio.h>

#define max 39

int ReadScore(int score[]);

int fun(int score[], int m, int below\_score[], int below\_index[]); //1

void main()

{

int a[max], n = -1;

int i;

int below\_score[max], below\_index[max], below\_n;

n = ReadScore(a); //1

below\_n = fun(a, n, below\_score, below\_index); //1

if (n == 0) //1

{

printf("there are no valid scores\n");

exit(0);

}

printf("the number of the class:%d\n", n);

printf("the number under the average score: %d\n", below\_n);

for (i = 0; i < below\_n; i++)

{

printf("the %dth score is: %d\n", below\_index[i] + 1, below\_score[i]); //1

}

}

int ReadScore(int score[])

{

int i = -1;

// printf("Input score:");

do

{

i++;

scanf("%d", &score[i]);

} while (score[i] >= 0); //1

return i;

}

int fun(int score[], int m, int below\_score[], int below\_index[])

{

int i, k = 0;

float aver = 0;

for (i = 0; i < m; i++)

aver += score[i]; //1

aver /= m; //1

for (i = 0; i < m; i++)

if (score[i] < aver)

{

below\_score[k] = score[i];//1

below\_index[k] = i;//1

k++;

}

return k;//1

}

填表格

将1、2、3、4、5和6填入2行3列的表中，要使得每一个列右边的数字比左边的数字大，每一行下面的数字比上面的数字大。按此要求，可有几种填写方法？

\*\*输出提示信息："The possible table satisfied above conditions are:\n"

\*\*输出格式要求："\nNo.:%d" "\n%3d" "%3d"

#include <stdio.h>

#include <stdlib.h>

#define N 6

int main()

{

int i, j, k, m, n, num;

int a[N];

a[0] = 1;

a[5] = 6;

num = 0;

printf("The possible table satisfied above conditions are:\n");

for (i = 2; i <= 5; i++)

for (j = 2; j <= 5; j++)

for (n = 2; n <= 5; n++)

for (m = 2; m <= 5; m++)

{

a[1] = i;

a[2] = j;

a[3] = n;

a[4] = m;

if (a[2] > a[1] && a[4] > a[3] && a[4] > a[1] && a[1] != a[3] && a[2] != a[3] && a[2] != a[4])

{

num++;

printf("\nNo.:%d", num);

for (k = 0; k < 6; k++)

{

if (k % 3 == 0)

printf("\n");

printf("%3d", a[k]);

}

}

}

return 0;

}

全部

【附加题】从键盘输入某单位职工的月收入(人数最多不超过40人)，当输入负值时，表示输入结束，编程从键盘任意输入一个职工号，查找该职工的月收入。

程序运行示例1：

Input person's ID and income:001 564↙

Input person's ID and income:002 365↙

Input person's ID and income:003 564↙

Input person's ID and income:004 654↙

Input person's ID and income:005 -9↙

Total number is 4

Input the searching ID:004↙

income = 654

程序运行示例2：

Input person's ID and income:001 234↙

Input person's ID and income:002 654↙

Input person's ID and income:003 897↙

Input person's ID and income:004 321↙

Input person's ID and income:005-7↙

Total number is 4

Input the searching ID:009↙

Not found!

下面给出的程序存在多处错误，请修正所有错误，使之能够得到正确的运行结果，并将正确程序填写在答题区。

#include <stdio.h>

#define N 40

int ReadScore(int income[], long num[]);

int BinSearch(long num[], long x, int n);

int main()

{

int income[N], n, pos;

long num[N], x;

n = ReadScore(income, num);

printf("Total number is %d\n", n);

printf("Input the searching ID:");

scanf("%d", &x);

pos = BinSearch(num, x, n);

if (pos != -1)

{

printf("income = %d\n", income[pos]);

}

else

{

printf("Not found!\n");

}

return 0;

}

int ReadScore(int income[], long num[])

{

int i ;

do

{

i++;

printf("Input person's ID and income:");

scanf("%d%d", num[i], income[i]);

} while (num[i] >0 || income[i] >= 0);

return i;

}

int BinSearch(long num[], long x, int n)

{

int low, high, mid;

low = 0;

high = n ;

while (low < high)

{

mid = (high + low) / 2;

if (x > num[mid])

{

low = mid + 1;

}

else if (x < num[mid])

{

high = mid - 1;

}

else

{

return (mid);

}

}

return(-1);

}

#include <stdio.h>

#define N 40

int ReadScore(int income[], long num[]);

int BinSearch(long num[], long x, int n);

int main()

{

int income[N], n, pos;

long num[N], x;

n = ReadScore(income, num);

printf("Total number is %d\n", n);

printf("Input the searching ID:");

scanf("%ld", &x);

pos = BinSearch(num, x, n);

if (pos != -1)

{

printf("income = %d\n", income[pos]);

}

else

{

printf("Not found!\n");

}

return 0;

}

int ReadScore(int income[], long num[])

{

int i = -1;

do {

i++;

printf("Input person's ID and income:");

scanf("%ld%d", &num[i], &income[i]);

} while (num[i] > 0 && income[i] >= 0);

return i;

}

int BinSearch(long num[], long x, int n)

{

int low, high, mid;

low = 0;

high = n - 1;

while (low <= high)

{

mid = (high + low) / 2;

if (x > num[mid])

{

low = mid + 1;

}

else if (x < num[mid])

{

high = mid - 1;

}

else

{

return (mid);

}

}

return(-1);

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include <stdio.h>

#define N 40

int ReadScore(int income[], long num[]);

int BinSearch(long num[], long x, int n);

int main()

{

int income[N], n, pos;

long num[N], x;

n = ReadScore(income, num);

printf("Total number is %d\n", n);

printf("Input the searching ID:");

scanf("%ld", &x);

pos = BinSearch(num, x, n);

if (pos != -1)

{

printf("income = %d\n", income[pos]);

}

else

{

printf("Not found!\n");

}

return 0;

}

int ReadScore(int income[], long num[])

{

int i = -1;

do{

i++;

printf("Input person's ID and income:");

scanf("%ld%d", &num[i], &income[i]);

} while (num[i] >0 && income[i] >= 0);

return i;

}

int BinSearch(long num[], long x, int n)

{

int low, high, mid;

low = 0;

high = n - 1;

while (low <= high)

{

mid = (high + low) / 2;

if (x > num[mid])

{

low = mid + 1;

}

else if (x < num[mid])

{

high = mid - 1;

}

else

{

return (mid);

}

}

return(-1);

}

给定如下定义：

struct date\_rec

{

int day ;

int month ;

int year ;

} ;

struct date\_rec current\_date ;

写一个程序包含如下的函数，完成：

(a) 输入current\_date的值：

void input\_date(struct date\_rec \*current\_date)

(b) 将current\_date增加1天：

void increment\_date(struct date\_rec \*current\_date)

(c) 显示current\_date的值：

void output\_date(struct date\_rec \*current\_date)

考虑每个月的实际天数，同时也考虑闰年的情况。

\*\*输入格式要求："%d%d%d" 提示信息："请输入当前日期（年 月 日）："

\*\*输出格式要求："当前日期：%d年%d月%d日！" （加1天后的新日期）

21根火柴游戏。现有21根火柴，两人轮流取，每人每次可以取1至4根，不可多取（假如多取或者取走的数量不在合法的范围内，则要求重新输入），也不能不取，谁取最后一根火柴谁输。请编写一个程序进行人机对弈，要求人先取，计算机后取；请设计一种计算机取走火柴的规则，使得计算机一方为常胜将军。

\*\*输入格式要求："%d" 提示信息："Game begin:\n" "How many sticks do you wish to take (1~%d)?"

\*\*输出格式要求：" %d sticks left in the pile.\n" " You have taken the last sticks.\n"

" \*\*\*You lose!\nGame Over.\n" "Computer take %d sticks.\n"

程序运行示例如下：

Game begin:

How many sticks do you wish to take (1~4)?6

How many sticks do you wish to take (1~4)?3

18 sticks left in the pile.

Computer take 2 sticks.

16 sticks left in the pile.

How many sticks do you wish to take (1~4)?3

13 sticks left in the pile.

Computer take 2 sticks.

11 sticks left in the pile.

How many sticks do you wish to take (1~4)?3

8 sticks left in the pile.

Computer take 2 sticks.

6 sticks left in the pile.

How many sticks do you wish to take (1~4)?3

3 sticks left in the pile.

Computer take 2 sticks.

1 sticks left in the pile.

How many sticks do you wish to take (1~1)?2

How many sticks do you wish to take (1~1)?1

You have taken the last sticks.

\*\*\*You lose!

Game Over.

#include<stdio.h>

main()

{int a=21,i;

printf("Game begin:\n");

while(a>0)

{do

{printf("How many sticks do you wish to take (1~%d)?",a>4?4:a);

scanf("%d",&i);

}while(i>4||i<1||i>a);

if(a-i>0)

printf(" %d sticks left in the pile.\n",a-i);

if((a-i)<=0)

{

printf(" You have taken the last sticks.\n");

printf(" \*\*\*You lose!\nGame Over.\n");

break;

}

else

printf("Computer take %d sticks.\n",5-i);

a-=5;

printf(" %d sticks left in the pile.\n",a);

}

}

求两个正整数的最大公约数。编写求最大公约数的函数Gcd()，主函数中输入任意两个整数，调用此函数求最大公约数并输出结果。

要求函数原型为：int Gcd(int a, int b)，a、b为输入的任意两个整数，当a或b为0或负数时，函数返回值为-1，否则返回a、b的最大公约数。

#include <stdio.h>

int Gcd(int a, int b);

int main()

{

int a, b, c;

printf("Input a,b:");//1

scanf("%d,%d", &a, &b);//2

c = Gcd(a, b); //2

if (c != -1)

{

printf("Greatest Common Divisor of %d and %d is %d\n", a, b, c);//1

}

else

{

printf("Input number should be positive!\n");//1

}

return 0;

}

int Gcd(int a, int b) //2

{

int i, t;

if (a <= 0 || b <= 0) //2

{

return -1;

}

t = a < b ? a : b; //1

for (i = t; i > 0; i--) //2

{

if (a % i == 0 && b % i == 0) return i; //2

}

return 1;

}

数组旋转：

编写程序将如下的一个固定大小的整数数组a[2][3]向右旋转90度，构成新的数组b[3][2]，并打印出来。

例如：a=1 2 3

4 5 6

旋转后

b=4 1

5 2

6 3

\*\*输入提示信息："Array a:\n"

\*\*输入格式要求："%d"

\*\*输出格式要求： "Array b:\n"

输出时每个元素的大小为"%4d"

程序运行示例如下：

Array a:

1 2 3

4 5 6

Array b:

4 1

5 2

6 3

#include <stdio.h>

int main()

{

int a[2][3], b[3][2];

int i, j;

printf("Array a:\n");

for (i = 0; i < 2; i++)

{

for (j = 0; j < 3; j++)

scanf("%d", &a[i][j]);

}

for (i = 0; i < 2; i++)

{

for (j = 0; j < 3; j++)

{

b[j][i] = a[i][j];

}

}

for (i = 0; i < 3; i++)

{

int tmp = b[i][0];

b[i][0] = b[i][1];

b[i][1] = tmp;

}

printf("Array b:\n");

for (i = 0; i < 3; i++)

{

for (j = 0; j < 2; j++)

{

printf("%4d", b[i][j]);

}

printf("\n");

}

return 0;

}

输入一个英文句子(输入句子以回车作为结束,句子长度包括标点符号并且不超过20)，翻转句子中单词的顺序，但单词内字符的顺序不变。标点符号和普通字母一样处理。例如输入字符串“I am a student.”，则输出“student.a am I”

输入提示信息：Input Sentence:

输出提示信息：Output Reverse Sentence:

#include "stdio.h"

#include "string.h"

int main()

{

char arr[20];

int i, j, k;

printf("Input Sentence:");

gets(arr);

i = strlen(arr) - 1;

k = i;

printf("Output Reverse Sentence:");

while (i >= 0)

{

while (arr[i] != ' ' && i >= 0)

{

i--;

}

for (j = i + 1; j <= k; j++)

{

printf("%c", arr[j]);

}

while (arr[i] == ' ' && i >= 0)

{

printf(" ");

i--;

}

k = i;

}

return 0;

}

输入字符串，包含：字母、数字、标点符号，以及空格字符，并将其逆序输出。

例如，当输入字符串为“I am a student.”,输出为“.tneduts a ma I”，假设字符数组最大长度为30。

输入提示信息："Please Enter String1:\n"

输入格式：gets()

输出格式："Result is:\n%s\n"

#include <stdio.h>

#include <string.h>

#define N 30

int main()

{

char str[N], c;

int n, i, j;

printf("Please Enter String1:\n");

gets(str); //1

n = strlen(str) - 1; //1

for (i = 0; i <= n / 2; i++) //2

{

c = str[i]; //1

str[i] = str[n - i]; //1

str[n - i] = c; //1

}

printf("Result is:\n%s\n", str);

return 0;

}

有100g药品，用天平称量，砝码只有1g, 2g, 5g三种，若要求加的砝码总数为30个，有几种不同的加法？用穷举法编程求解。

\*\*输入提示信息和格式：无

\*\*输出格式："%2d,%2d,%2d\n"

注：（1）不能使用指针、结构体、共用体、文件、goto、枚举类型进行编程。

用纯C语言编程，所有变量必须在第一条可执行语句前定义。

#include <stdio.h>

int main()

{

int i,j,k,sum;

sum=0;//2

for(i=0;i<30;i++)//2

{

for(j=0;j<30;j++)//2

{

k=30-i-j;//2

if(i+2\*j+5\*k==100)//2

{

sum++;//2

printf("%2d,%2d,%2d\n",sum,i,j,k);//2

}

}

}

return 0;

}

Two Bags of Potatoes

time limit per test1 second

memory limit per test256 megabytes

inputstandard input

outputstandard output

Valera had two bags of potatoes, the first of these bags contains x (x≥1)

potatoes, and the second — y (y≥1) potatoes. Valera — very scattered

boy, so the first bag of potatoes (it contains x potatoes) Valera lost.

Valera remembers that the total amount of potatoes (x+y) in the two bags,

firstly, was not gerater than n, and, secondly, was divisible by k.

Help Valera to determine how many potatoes could be in the first bag.

Print all such possible numbers in ascending order.

Input

The first line of input contains three integers y, k, n (1≤y,k,n≤10^9;

n/k≤10^5).

Output

Print the list of whitespace-separated integers — all possible values

of x in ascending order. You should print each possible value of x exactly

once. If there are no such values of x print a single integer -1.

#include<stdio.h>

int main()

{

int y,n,k,i,j,m,y1,k1;

scanf("%d %d %d",&y,&k,&n);

y1=(y/k+1)\*k;

if(y1<=n)

{

while(y1<=n)

{

while(y1-y<=0)y1+=k;

printf("%d ",y1-y);

y1+=k;

}

}

else printf("-1");

return 0;

}

编程计算a%1 + aa%2 + aaa%3 + ... + aa...a%n(最后一项是n个a对n求余)

的值，然后输出这个值。

（要求存储累加项及总和的变量定义为长整型,a与n定义为整型），

其中n和a的值由键盘输入。

\*\*要求：

\*\*输入提示信息格式为："Enter n,a:\n"

\*\*输入格式为："%d,%d"

\*\*输出格式为："Sum=%ld\n"

\*\*请严格按照以上要求输入输出，除了以上指定输出结果外，

不允许有其他临时输出

#include <stdio.h>

main()

{

long term = 0, sum = 0;//2

int a, i, n;

printf("Enter n,a:\n");

scanf("%d,%d", &n, &a); //2

for (i = 1; i <= n; i++) //3

{

term = term \* 10 + a; //3

sum = sum + term % i; //3

}

printf("Sum=%ld\n", sum); //1

}

请安如下函数编程实现将一个字符串内所有的小写字母转换成大写字母，字符串由大、小写英文字母、数字、空格等构成。

函数原型：

void Change(char string[])

输入提示信息："please input a string:"

输出提示信息及格式："changed string is %s\n"

<友情提示>

小写字母和大写字母的ASCII码值之间的差是32

#include <string.h>

void Change(char string[])

{

int len = strlen(string);

int i = 0, j = 0;

for (i = 0; i < len; i++) //1

{

if (string[i] >= 'a' && string[i] <= 'z') //1

{

string[i] = string[i] - 'a' + 'A'; //1

}

}

}

int main()

{

char string[100];

printf("please input a string:"); //1

gets(string); //1

Change(string); //1

printf("changed string is %s\n", string); //1

return 0;

}

在主函数中从键盘输入某班学生某门课程的成绩(已知班级人数最多不超过40人，具体人数由键盘输入),试编程计算其平均分，并计算出成绩高于平均分的学生的人数。

要求：调用函数aver()，计算n名学生的平均成绩返回给主函数，然后在主函数中输出学生的平均成绩。

函数原型：float aver(int score[],int n);

\*\*\*输入提示信息：无

\*\*\*输入格式：输入学生人数用"%d"，输入学生成绩用"%d"

\*\*\*输出平均成绩提示信息和格式："Average score is %10.2f\n"

\*\*\*输出平均分以上的学生人数提示信息和格式:"The number of students in more than average %d\n"

注：（1）不能使用指针、结构体、共用体、文件、goto、枚举类型进行编程。

（2）用纯C语言编程，所有变量必须在第一条可执行语句前定义。

#include <stdio.h>

#define N 30

float Average(int score[],int n); //1

int main()

{

int score[N];

float aver;

int n,i,counter=0;

scanf("%d",&n);//1

for(i=0;i<n;i++)

{

scanf("%d",&score[i]); //1

}

aver=Average(score,n); //1

printf("Average score is %10.2f\n",aver);//1

for(i=0;i<n;i++)

{

if(score[i]>aver)counter++; //1

}

printf("The number of students in more than average %d\n",counter);

return 0;

}

float Average(int score[],int n)

{

int i;

int sum=0; //1

for(i=0;i<n;i++)

{

sum+=score[i]; //1

}

return (float)sum/n; //1

}

编写一个简单的23 根火柴游戏程序，实现人跟计算机玩这个游戏的程序。

为了方便程序自动评测，假设计算机移动的火柴数不是随机的，而是将剩余的火柴根数对3求余后再加1来作为计算机每次取走的火柴数。如果计算机打算移走的火柴数等于剩下的火柴数，则将计算机打算移走的火柴数减1。但是计算机不可以不取，剩下的火柴数为1时，必须取走1根火柴。

游戏规则是：

1、两个游戏者开始拥有23 根火柴棒；

2、每个游戏者轮流移走1 根、2 根或3 根火柴；

3、谁取走最后一根火柴为失败者。

玩家赢得运行示例如下：

这里是23根火柴游戏！！

注意：最大移动火柴数目为三根

请输入您移动的火柴数目：

2

您移动的火柴数目为：2

您移动后剩下的火柴数目为：21

计算机移动的火柴数目为：1

计算机移动后剩下的火柴数目为：20

请输入您移动的火柴数目：

2

您移动的火柴数目为：2

您移动后剩下的火柴数目为：18

计算机移动的火柴数目为：1

计算机移动后剩下的火柴数目为：17

请输入您移动的火柴数目：

2

您移动的火柴数目为：2

您移动后剩下的火柴数目为：15

计算机移动的火柴数目为：1

计算机移动后剩下的火柴数目为：14

请输入您移动的火柴数目：

2

您移动的火柴数目为：2

您移动后剩下的火柴数目为：12

计算机移动的火柴数目为：1

计算机移动后剩下的火柴数目为：11

请输入您移动的火柴数目：

2

您移动的火柴数目为：2

您移动后剩下的火柴数目为：9

计算机移动的火柴数目为：1

计算机移动后剩下的火柴数目为：8

请输入您移动的火柴数目：

2

您移动的火柴数目为：2

您移动后剩下的火柴数目为：6

计算机移动的火柴数目为：1

计算机移动后剩下的火柴数目为：5

请输入您移动的火柴数目：

1

您移动的火柴数目为：1

您移动后剩下的火柴数目为：4

计算机移动的火柴数目为：2

计算机移动后剩下的火柴数目为：2

请输入您移动的火柴数目：

1

您移动的火柴数目为：1

您移动后剩下的火柴数目为：1

计算机移动的火柴数目为：1

计算机移动后剩下的火柴数目为：0

恭喜您！您赢了！

玩家输的运行示例如下：

这里是23根火柴游戏！！

注意：最大移动火柴数目为三根

请输入您移动的火柴数目：

3

您移动的火柴数目为：3

您移动后剩下的火柴数目为：20

计算机移动的火柴数目为：3

计算机移动后剩下的火柴数目为：17

请输入您移动的火柴数目：

3

您移动的火柴数目为：3

您移动后剩下的火柴数目为：14

计算机移动的火柴数目为：3

计算机移动后剩下的火柴数目为：11

请输入您移动的火柴数目：

3

您移动的火柴数目为：3

您移动后剩下的火柴数目为：8

计算机移动的火柴数目为：3

计算机移动后剩下的火柴数目为：5

请输入您移动的火柴数目：

2

您移动的火柴数目为：2

您移动后剩下的火柴数目为：3

计算机移动的火柴数目为：2

计算机移动后剩下的火柴数目为：1

请输入您移动的火柴数目：

1

您移动的火柴数目为：1

您移动后剩下的火柴数目为：0

对不起！您输了！

补充说明：如果输入的火柴数超过3，则输出"对不起！您输入了不合适的数目，请点击任意键重新输入！\n"，如果玩家赢了，则输出"恭喜您！您赢了！ \n"

#include<stdio.h>

int main()

{

int g = 23;

int k = 3;

int b, c;

printf("这里是23根火柴游戏！！\n");

printf("注意：最大移动火柴数目为三根\n");

do

{

printf("请输入您移动的火柴数目：\n");

scanf("%d", &b);

if (b < 1 || b > 3)

{

printf("对不起！您输入了不合适的数目，请点击任意键重新输入！\n");

}

else

{

g = g - b;

printf("您移动的火柴数目为：%d\n", b);

printf("您移动后剩下的火柴数目为：%d\n", g);

if (g <= 0)

{

printf("对不起！您输了！\n");

break;

}

else

{

if (g <= 3 && g != 1)

{

c = g - 1;

}

else if (g == 1)

{

c = 1;

}

else

{

c = g % k + 1;

}

g = g - c;

printf("计算机移动的火柴数目为：%d\n", c);

printf("计算机移动后剩下的火柴数目为：%d\n", g);

if (g <= 0)

{

printf("恭喜您！您赢了！ \n");

break;

}

}

}

}

while (g > 0);

return 0;

}

输入10个数，将10个整数按升序排列输出，并且奇数在前，偶数在后。如果输入的10个数是:10 9 8 7 6 5 4 3 2 1 ，则输出：1 3 5 7 8 2 4 6 8 10。（编程提示：可利用2个数组变量，一个用来存放输入的整数，输入后，对这个数组进行排序，然后将数据复制到另一个数组中，先复制奇数再复制偶数）。

输入提示：Input 10 numbers:

输入格式：%d

输出格式：Output：%d,%d,%d,%d,%d,%d,%d,%d,%d,%d

#include "stdio.h"

int main()

{

int arr[10], brr[10];

int i, j, tmp;

printf("Input 10 numbers:");

for (i = 0; i < 10; i++)

{

scanf("%d", &arr[i]);

}

for (i = 0; i < 9; i++)

{

for (j = i + 1; j < 10; j++)

{

if (arr[j] < arr[i])

{

tmp = arr[i];

arr[i] = arr[j];

arr[j] = tmp;

}

}

}

for (i = 0, j = 0; i < 10; i++)

{

if (arr[i] % 2 == 1)

{

brr[j++] = arr[i];

}

}

for (i = 0; i < 10; i++)

{

if (arr[i] % 2 == 0)

{

brr[j++] = arr[i];

}

}

printf("Output: ");

for (i = 0; i < 10; i++)

{

if (i < 9)

{

printf("%d,", brr[i]);

}

else

{

printf("%d\n", brr[i]);

}

}

}

窗体顶端

程序模拟简单运算器的工作：输入一个算式（没有空格），遇等号'='说明输入结束，输出结果。假设计算器只能进行加、减、乘、除运算，运算数和结果都是整数，4种运算符的优先级相同，按从左到右的顺序计算。例如输入"1+2\*10-10/2="后，输出10。

\*\*输出格式要求："%d" 出错提示信息："错误的运算符：%c"

程序运行示例如下：

1+2\*10-10/2= <===此行为输入

10 <===此行为输出

窗体底端

#include <stdio.h>

int main(void)

{

int num, num1, op;

char ch;

num = 0;

num1 = 0;

op = '+';

do

{

ch = getchar();

if (ch >= '0' && ch <= '9')

{

num1 = num1 \* 10 + (ch - '0');

}

else

{

if (op == '+')

{

num = num + num1;

}

else if (op == '-')

{

num = num - num1;

}

else if (op == '\*')

{

num = num \* num1;

}

else if (op == '/')

{

num = num / num1;

}

else

{

printf("错误的运算符：%c", op);

exit(0);

}

op = ch;

num1 = 0;

}

} while (ch != '=');

printf("%d", num);

return 0;

}

窗体顶端

猴子吃桃程序\_扩展2

猴子第一天摘了若干个桃子，吃了一半，不过瘾，又多吃了1个。第二天早上将剩余的桃子又吃掉一半，并且又多吃了1个。此后每天都是吃掉前一天剩下的一半零一个。到第n天再想吃时，发现只剩下1个桃子，问第一天它摘了多少桃子？为了加强交互性，由用户输入不同的天数n进行递推，即假设第n天的桃子数为1。同时还要增加对用户输入数据的合法性验证（如:不允许输入的天数是0和负数）

程序运行结果示例：

Input days:

0↙

Input days:

-5↙

Input days:

a↙

Input days:

3↙

x=10

输入格式:"%d"

输出格式：

输入提示信息："Input days:\n"

输出："x=%d\n"

窗体底端

#include<stdio.h>

int main()

{

int x = 1, day, n, flag;

do

{

flag = 0;

printf("Input days:\n");

n = scanf("%d", &day);

if (n != 1)

{

flag = 1;

while (getchar() != '\n');

}

if (day <= 0)

flag = 1;

} while (flag == 1);

while (day > 1)

{

x = (x + 1) \* 2;

day--;

}

printf("x=%d\n", x);

return 0;

}

程序改错-3

从键盘任意输入两个符号各异的整数，直到输入的两个整数满足要求为止，然后打印这两个数。请通过测试找出下面这个程序存在的问题（不止一个问题哦），并改正。同时用下面给出的运行结果示例检查修改后的程序。

#include <stdio.h>

int main()

{

int x1, x2;

do

{

printf("Input x1, x2:");

scanf("%d,%d", &x1, &x2);

}

while (x1 \* x2 > 0);

printf("x1=%d,x2=%d\n", x1, x2);

return 0;

}

程序正确的运行结果示例：

Input x1, x2:

a,s↙

Input x1, x2:

a,1↙

Input x1, x2:

2,s↙

Input x1, x2:

1,2↙

Input x1, x2:

-1,-2↙

Input x1, x2:

0,3↙

Input x1, x2:

1.2,3.4↙

Input x1, x2:

1.2,5↙

Input x1, x2:

-1,3↙

x1=-1,x2=3

输入格式: "%d,%d"

输入提示信息："Input x1, x2:\n"

输出格式："x1=%d,x2=%d\n"

#include <stdio.h>

int main()

{

int x1, x2, n, flag;

do

{

flag = 0;

printf("Input x1, x2:\n");

n = scanf("%d,%d", &x1, &x2);

switch (n)

{

case 0:

while (getchar() != '\n');

flag = 1;

break;

case 1:

while (getchar() != '\n');

flag = 1;

break;

default:

if (x1 \* x2 >= 0)

flag = 1;

else

flag = 0;

break;

}

}

while (flag == 1);

printf("x1=%d,x2=%d\n", x1, x2);

return 0;

}