**1.链表一拆三**

#include<stdio.h>

#include<stdlib.h>

#pragma pack(1)

struct student

{int num;

char name[10];

int grade;

};

struct stunode

{struct student \*data;

struct stunode \*next;

};

void add(struct student \* m,struct stunode \*\*n,struct stunode \*\*head)

{ struct stunode \*p;

p = (struct stunode \*)malloc(sizeof(struct stunode));

if (p == NULL)

{ printf("can’t get memory!\n");

exit(1);

}

p->data =m;p->next=NULL;

if (\*head== NULL)

\*head = p;

else

(\*n)->next=p;

\*n=p;

}

void outpoly(struct stunode \*head)

{ struct stunode \*p;

struct student \*k;

p=head;

printf("学号（num）姓名（name） 成绩（grade）\n");

if(p==NULL){ printf("error!\n"); return; }

do

{ k=p->data;

printf("%02d %s %3d\n",(\*k).num,(\*k).name,(\*k).grade);

p=p->next;

}while(p!=NULL);

}

void frepoly(struct stunode \*head)

{ struct stunode \*k,\*p;

k=head;

if(k!=NULL)

{p=k; k=k->next;free(p);}

}

main()

{ int i,k,flag=0;

struct student \*a,\*p[3];

struct stunode \*list[3],\*head[3];

a=(struct student \*)malloc(sizeof(struct student)\*3);

if (a == NULL)

{ printf("can’t get memory!\n");

exit(1);

}

for(i=0;i<3;i++)

{ list[i] = NULL;

head[i] = NULL;

}

for(i=0;i<3;i++)

{ k=i;

printf("input num,name and grade:\n");

scanf("%d%s%d", &a[k].num, a[k].name,&a[k].grade);

flag=a[k].grade/10-7;

if(flag==3)flag=flag-1;

p[k]=&a[k];

add(p[k],&list[flag],&head[flag]);

}

for(i=0;i<3;i++)

outpoly(head[i]);

for(i=0;i<3;i++)

frepoly(head[i]);

free(a);

}

**2.围圈报数**

#include<stdio.h>

#include<stdlib.h>

struct NODE

{int data;

struct NODE \*next;

};

main()

{int m,n,i;

struct NODE \*node=NULL,\*head=NULL,\*end=NULL;

printf("input m,n\n");

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

head=(struct NODE \*)malloc(sizeof(struct NODE));

head->data=1;

head->next=NULL;

end=head;

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

{node=(struct NODE \*)malloc(sizeof(struct NODE));

node->data=i;

node->next=NULL;

end->next=node;

end=node;

}

end->next=head;

end=head;

while(end->next!=end){

for(i=1;i<n;i++){node=end;end=end->next;}

node->next=end->next;

end=end->next;

}

printf("%d\n",end->data);

}

**3.字符串替换**

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

void replace(char \*s1,char \*s2,char \*s3)

{int length;

char m[10000];

char \*p=strstr(s1,s2);

while (p)

{memset(m,0,10000);

length=p-s1;

strncpy(m,s1,length);/\*将原字符串s2之前的内容复制到m\*/

strcat(m,s3); /\*把s2换成s3\*/

strcat(m,p+strlen(s2));/\*把s2之后的字符串复制回m\*/

strcpy(s1,m);

p=strstr(s1,s2);}

}

**4.大数乘法**

#include<stdio.h>

#include<string.h>

#define N 10000

main()

{char a[N],b[N];

int A[N],B[N],c[N];

int i,l1,l2,l3,j;

printf("input a\n");

scanf("%s",&a);

l1=strlen(a);

for(i=0;i<l1;i++)A[i]=a[i]-48;

printf("input b\n");

scanf("%s",&b);

l2=strlen(b);

for(i=0;i<l2;i++)B[i]=b[i]-48;

for(i=0;i<l1+l2;i++)c[i]=0;

for(i=0;i<l1;i++){

for(j=0;j<l2;j++)

c[i+j]=c[i+j]+A[i]\*B[j];

}

l3=l2+l1-1;

for(i=l3-1;i>0;i--){

c[i-1]=c[i-1]+c[i]/10;

c[i]=c[i]%10;

}

printf("a\*b=");

for(i=0;i<l3;i++)printf("%d",c[i]);

printf("\n");

}

**5.全排列**

#include<stdio.h>

#include<stdlib.h>

void sort(int \*a,int n,int k)

{int i,temp;

if(k==n-1)

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

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

printf("\n");

}

else{for(i=k;i<n;i++)

{temp=a[k];

a[k]=a[i];

a[i]=temp;

sort(a,n,k+1);

temp = a[k];

a[k] = a[i];

a[i] = temp;

}}

}

main()

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

sort(a,5,0);

}

**6.数二**

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

#include<math.h>

main()

{int count=0,i,N,l;

int a,b,c;

printf("input N\n");

scanf("%d",&N);

l=(int)log10((double)N)+1;

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

{a=N%(int)pow(10,i-1);

c=N/(int)pow(10,i);

b=N/(int)pow(10,i-1)%10;

if(b>1)count=count+c\*(int)pow(10,i-1)+a+1;

if(b<2)count=count+c\*(int)pow(10,i-1);

}

count\*=2;

printf("the number of 2 is %d\n",count);

}

**7.加减法**

include<stdio.h>

#include<stdlib.h>

#include<string.h>

main()

{char s[100],\*\*a,\*sym;

int i,j,part=1,num,place=0,count,n,sum=0;

int \*b;

printf("input the arithmetic formula\n");

gets(s);

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

{if(s[i]=='+'||s[i]=='-')part=part+1;}

a=(char \*\*)malloc(sizeof(char\*)\*part);

sym=(char \*)malloc(sizeof(char)\*(part-1));

b=(int \*)malloc(sizeof(int)\*part);

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

{count=0;n=place;

for(j=place;j<strlen(s);j++)

{if(s[j]>='0'&&s[j]<='9'){count++;place++;}

else break;}

if(i<part-1)sym[i]=s[place];

place++;

a[i]=(char \*)malloc(sizeof(char)\*(count+1));

strncpy(a[i],s+n,count);

(a[i])[count]='\0';

}

for(i=0;i<part;i++){b[i]=atoi(a[i]);free(a[i]);}

sum=b[0];

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

{if(sym[i]=='+')sum=sum+b[i+1];

if(sym[i]=='-')sum=sum-b[i+1];

}

printf("%d\n",sum);

free(b);free(sym);

free(a);}

**8.排序（小到大）**

void sort(char \*c,int n)

{int i,j;

char k;

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

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

{if(c[j]>c[j+1]){k=c[j];c[j]=c[j+1];c[j+1]=k;}

}}

}

**9.伴随矩阵（转置与行列式）**

#include<stdio.h>

#include<stdlib.h>

void T(int a[ ],int b[ ],int n,int i)

{

int p,q,m;

m=0;

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

if (p!=i)

{

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

b[m\*(n-1)+q]=a[p\*n+q+1];

m++;

}

}

int D(int a[],int n)

{

int i,j,s,\*b;

i=0;

j=0;

s=0;

if (n==2) s=(a[i\*n+j]\*a[(i+1)\*n+j+1])-(a[i\*n+j+1]\*a[(i+1)\*n+j]);

else

{

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

{

b=(int\*)malloc((n-1)\*(n-1)\*sizeof(int));

T(a,b,n,i);

if (i%2==0)

s=s+a[i\*n]\*D(b,n-1);

else s=s-a[i\*n]\*D(b,n-1);

free(b);

}

}

return(s);

}

int C(int a[],int n,int i,int j)

{

int \*b;

int p,q,m,k,s;

b=(int\*)malloc((n-1)\*(n-1)\*sizeof(int));

m=0;

k=0;

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

if (p!=i)

{

k=0;

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

if (q!=j)

{

b[m\*(n-1)+k]=a[p\*n+q];

k++;

}

m++;

}

s=D(b,n-1);

return(s);

}

main()

{

int i,j,n,\*a,\*c;

void T(int a[],int b[],int n,int i);

int D(int a[],int n);

int C(int a[],int n,int i,int j);

printf("please input n=");

scanf("%d",&n);

a=(int\*)malloc(n\*n\*sizeof(int));

c=(int\*)malloc(n\*n\*sizeof(int));

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

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

{

printf("%drow %dcolumn:",i+1,j+1);

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

}

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

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

{

if ((i+j)%2==0)

c[i\*n+j]=C(a,n,j,i);

else c[i\*n+j]=-C(a,n,j,i);

}

printf("the adj matrix:\n");

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

{

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

printf("%d ",c[i\*n+j]);

printf("\n");

}

free(c);

free(a);

}

**10.子集对半拆分**

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

static int s=0;

int devide(int \*a,int n,int half)

{int i,j,sum=0,stop=0;

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

{sum=sum+a[i];

if(sum==half){stop=1;break;}

}

if(stop==1)

{printf("TRUE\n");

for(j=0;j<=i;j++)printf("%d ",a[j]);

printf("\n");

for(j=i+1;j<n;j++)printf("%d ",a[j]);

printf("\n");

}

return stop;

}

void order(int \*a,int n,int k,int half)

{int i,temp;

if(k==n-1)

{s=devide(a,n,half);

if(s==1) {free(a);exit(0);}

}

else

{for(i=k;i<n;i++)

{temp=a[k];a[k]=a[i];a[i]=temp;

order(a,n,k+1,half);

temp=a[k];a[k]=a[i];a[i]=temp;}

}

}

main()

{int \*a,n,i,half,sum=0;

printf("input the size of a\n");

scanf("%d",&n);

a=(int \*)malloc(sizeof(int)\*n);

printf("input %d numbers\n",n);

for(i=0;i<n;i++){scanf("%d",&a[i]);sum=sum+a[i];}

if(sum%2!=0){printf("False\n");free(a);exit(0);}

else half=sum/2;

order(a,n,0,half);

if(s==0)printf("False\n");

free(a);

}

**11.数字的不重复组合**

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

void swap(char \*a,char \*b)

{char k;

k=\*a;\*a=\*b;\*b=k;

}

void sort(char \*a,int n,int k,char\*\*p) /\*将a中数字的全排列储存进p中\*/

{int i;

static int j=0;

if(k==n-1)

{ strcpy(p[j],a);

j+=1;

}

else{for(i=k;i<n;i++)

{swap(&a[k],&a[i]);

sort(a,n,k+1,p);

swap(&a[k],&a[i]);

}}

}

main()

{char num[100],\*\*p,s[100],k[18];

int i=0,j=0,n,size=1,count;

gets(s);

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

{if(s[i]<='9'&&s[i]>'0')num[j++]=s[i];

if(s[i]=='0'){num[j]='\0';break;}

}

n=strlen(num);

for(i=1;i<=n;i++)size\*=i;

p=(char\*\*)malloc(sizeof(char\*)\*size);

for(i=0;i<size;i++)p[i]=(char \*)malloc(sizeof(char)\*(n+1));

count=size;

sort(num,n,0,p);

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

{for(j=i+1;j<size;j++)

{if(strcmp(p[j],p[i])>0)

{strcpy(k,p[i]);strcpy(p[i],p[j]);strcpy(p[j],k);

}}

}

puts(p[0]);

for(i=1;i<size;i++){if(strcmp(p[i],p[i-1])!=0)puts(p[i]);else count=count-1;}

for(i=1;i<size;i++)free(p[i]);

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

free(p);

}

**12.矩阵乘法**

#include<stdio.h>

#include<stdlib.h>

void T(int \*a,int \*b,int m,int n)

{int i,j,k;

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

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

{b[m\*i+j]=a[n\*j+i];

}}

}

main()

{int m,n,p;

int \*A,\*B,\*C,\*D;

int i,j,k;

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

A=(int \*)malloc(sizeof(int)\*m\*n);

B=(int \*)malloc(sizeof(int)\*n\*p);

C=(int \*)malloc(sizeof(int)\*m\*p);

D=(int \*)malloc(sizeof(int)\*p\*m);

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

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

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

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

{for(j=0;j<p;j++)

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

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

{for(j=0;j<p;j++)

{C[i\*p+j]=0;

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

{C[i\*p+j]+=A[i\*n+k]\*B[k\*p+j];}}

}

T(C,D,m,p);

for(i=0;i<p;i++){

for(j=0;j<m;j++)printf("%d ",D[i\*m+j]);

printf("\n");

}

free(A);free(B);free(C);free(D);

}

**13.局部反序去零**

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

main()

{char ip[16],temp[4];

int i,j=0,count=0;

printf("input the IP\n");

gets(ip);

for(i=0;i<16;i++)

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

temp[j++]=ip[i];

else{count=0;

for(j=2;j>=0;j--)

{if(temp[j]=='0'&&count==0);

else{count=1;putchar(temp[j]);}

}

putchar(ip[i]);

j=0;

}

}

}

**14.高斯消元**

#include <stdio.h>

#include <stdlib.h>

#define MAX 10

​

double A[MAX][MAX];

double b[MAX];

double X[MAX];

int NUM;

​

void Input\_Matrix()//输入矩阵

{

int i,j;

printf("系数矩阵A的阶数:\n");

scanf("%d",&NUM);

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

{

printf("系数矩阵A的第%d行元素:\n",i);

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

scanf("%lf",&A[i-1][j-1]);

}

printf("右端项b:\n");

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

{

scanf("%lf",&b[i-1]);

}

printf("输入的系数矩阵A:\n");

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

{

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

printf("%.4lf\t",A[i][j]);

printf("\n");

}

printf("输入的右端项b:\n");

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

printf("%.4lf\n",b[i]);

}

​

​

int main()

{

int i,j,k;

float mik;//消元过程所用变量

float S;//回代过程所用变量

Input\_Matrix();

//消元

for(k=0; k<NUM-1; k++)

{

if(!A[k][k])

return -1;

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

{

mik=A[i][k]/A[k][k];

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

{

A[i][j]=A[i][j]-mik\*A[k][j];

}

b[i]=b[i]-mik\*b[k];

}

}

printf("消元后的矩阵A:\n\n");

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

{

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

printf("%.4lf\t",A[i][j]);

printf("\n");

}

printf("消元后的右端项b:\n\n");

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

printf("%.4lf\n",b[i]);

//回代

X[NUM-1]=b[NUM-1]/A[NUM-1][NUM-1];

for(k=NUM-2; k>=0; k--)

{

S=b[k];

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

{

S=S-A[k][j]\*X[j];

}

X[k]=S/A[k][k];

}

printf("结果X=\n\n");

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

printf("%.4lf\n",X[i]);

return 0;

}