TUYỂN TẬP 1 SỐ ĐỀ THI VÀ CODE CÁC KỲ THI OLP TIN HỌC SV TOÀN QUỐC

1. CĐ 2005 : Dự trự nước Code : hienclubvn ; vietduc ; AlexBlack
2. CĐ 2005 : Địa đạo Code : hienclubvn ; vietduc; AlexBlack; Sounj
3. KC2005 : Tìm đặc trưng ảnh Code: hienclubvn; vietduc; AlexBlack
4. KC2005: Thám hiểm Code: AlexBlack

5. KC2006: Radar Code: hienclubvn; panaturo; AlexBlack
6. CĐ2006: Siêu mã Code: vietduc
7. Tập thể KC: Tính điểm Code: vietduc ; hienclubvn
8. Tập thể KC: Phân phòng ở Code: hienclubvn ; vietduc

9. KC2008 : Dãy số Code: hienclubvn

10. KC2009 : Đào tạo từ xa Code: vietduc
11. KC2009 : Dãy số Code: Sounj ; hunterphu
12. KC2009: Kết bạn Code: hienclubvn; Vibzz90
13. KC2009: Hiệu chỉnh ảnh đơn sắc Code: hienclubvn

Bài : Dự trữ nước (Cao đẳng 2005)

Ở miền Trung thường năm nào cũng có những đợt hạn hán nên ông Nam có những thùng dự trữ nước. Do mua làm nhiều đợt nên N ($1 \le N \le 1000$) thùng chứa nước của ông Nam có kích thước khác nhau, mỗi thùng có sức chứa Ci ($1 \le Ci \le 10000$, $1 \le i \le N$). Dự đoán rằng năm nay sẽ có đợt hạn hán lớn nên ông Nam muốn đổ đầy nước hết các thùng để dự trữ. Sau khi kiểm tra ông Nam thấy rằng có một số thùng vẫn còn đầy, một số khác thì vơi đi một phần, còn một số thì đã hết. Ông quyết định các thùng nào chưa đầy thì sẽ chở đi để đổ đầy nước. Nhưng do nơi lấy nước rất xa, và mỗi lần chỉ chở đi được 1 thùng nên ông quyết định sẽ san nước giữa các thùng với nhau để số thùng phải chở đi là ít nhất

Yêu cầu:

Cho dung lượng nước hiện có của thùng thứ i là Bi ($0 \le Bi \le Ci$, $1 \le i \le N$), hãy giúp ông Nam xác định số lượng thùng ít nhất phải mang đi.

Dữ liệu: vào từ file văn bản WATER.INP có dạng sau:

• Dòng thứ nhất ghi một số tự nhiên N là số lượng các thùng nước.

• Dòng thứ i trong N dòng tiếp theo mỗi dòng có 2 số nguyên Bi và Ci ($0 \le Bi \le Ci$) mô tả thông tin thùng thứ i, với Bi là nước còn trong thùng và Ci là sức
chứa của thùng, các số cách nhau ít nhất một khoảng trắng.

Kết quả: ghi ra file văn bản WATER.OUT chứa một số là số lượng ít nhất các
thùng nước tìm được.

Kết quả: ghi ra file văn bản WATER.OUT chứa một số là số lượng ít nhất các
thùng nước tìm được.

Ví du:

WATER.INP

4
0 1
4 5
0 2
1 2

WATER.OUT

1

- Ý tưởng : Bài này đi tìm số thùng chưa đầy (hoặc hết), để đem đi
  Vậy để số thùng mang đi là nhỏ nhất thì phải ưu tiên mang cái lớn trước
  Giải thuật: sắp xếp thứ tự tăng dần
  Lấy tổng lượng nước đổ đầy các thùng theo thứ tự từ nhỏ lên lớn, cho đến khi hết. Kiểm tra xem thùng nào chưa đầy thì vác đi

```
// Code của @hienclubvn
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#define input "WATER.INP"
#define output "WATER.OUT"
void Swap(int &a,int &b)
{
     a^=b^=a^b;
}
void Sort(int a[],int n)
{
     for(int i=0;i<n-1;i++)
       for(int j=i+1;j<n;j++)
       if (a[i]<a[j]) Swap(a[i],a[j]);
}
int main ()
{
    FILE *f,*f1;
    int N,i=0,Sum=0,*Arr,temp;
    f=fopen(input,"rt");
    fscanf(f,"%d\n",&N);
    Arr=(int*)malloc(N*sizeof(int));
    while(!feof(f))
    {
        fscanf(f,"%d ",&temp);
        fscanf(f,"%d\n",&Arr[i++]);
        Sum+=temp;
    }
    // Sap xep lai
    Sort(Arr,N);
    // thuc hien do day nuoc
    i=0;
    while(Sum&&i<N)
    {
        Sum-=Arr[i++];
```

```
            if (Sum<0) temp=N-i-1;
        }
        f1=fopen(output,"wt");
        fprintf(f1,"%d",temp);
        getch();
}
```

```
// Code của @vietduc
#include<iostream>
#include<fstream>
using namespace std;
int main()
{
    ifstream infile("water.txt");
    int water=0,n,*c,i,j;
    infile >> n;
    c=new int[n];
    for ( i=0; i<n; i++)
    {
        int a;
        infile >> a >> c[i];
        water+=a;
    }
    for (i=0; i<n-1; i++)
        for (j=i; j<n; j++)
            if (c[i]>c[j])
            {
                c[i]+=c[j];
                c[j]=c[i]-c[j];
                c[i]-=c[j];
            }
    i=0;
    while (water>=c[i])
        water-=c[i++];
    cout << n-i;
    return 0;
}
```

```
// code của @AlexBlack
#include<stdio.h>
#define input "water.txt"
#define MAX 100
typedef struct thung
{
    int dungtich;
    int luongnuoc;
};

int Nhap(thung th[],int &n)
{
    FILE *f=fopen(input,"rt");
    if(f==NULL)
        return 0;
    thung t1;
```

```
    fscanf(f,"%d",&n);
    for(int i=0;i<n;i++)
    {
        fscanf(f,"%d",&t1.luongnuoc);
        fscanf(f,"%d",&t1.dungtich);
        th[i]=t1;
    }
    fclose(f);
    return 1;
}

void Sort(thung th[],int n)
{
    for(int i=0;i<n-1;i++)
    {
        for(int j=i+1;j<n;j++)
        {
            if(th[i].dungtich>th[j].dungtich)
            {
                thung t1=th[i];
                th[i]=th[j];
                th[j]=t1;
            }
        }
    }
}

void DoNuoc(thung th[],int n)
{
    int j=0;
    for(int i=n-1;i>-1;i--)
    {
        if(th[i].luongnuoc!=0)
        {
            while(j<i)
            {
                if(th[j].luongnuoc!=th[j].dungtich)
                {
                    if((th[i].luongnuoc+th[j].luongnuoc)<=th[j].dungtich)
                    {
                        th[j].luongnuoc+=th[i].luongnuoc;
                        th[i].luongnuoc=0;
                        j++;
                        break;
                    }
                    else
                    {
                        int k=th[j].dungtich-th[j].luongnuoc;
                        th[j].luongnuoc=th[j].dungtich;
                        th[i].luongnuoc-=k;
                    }
                }
                j++;
            }
        }
    }
}
```

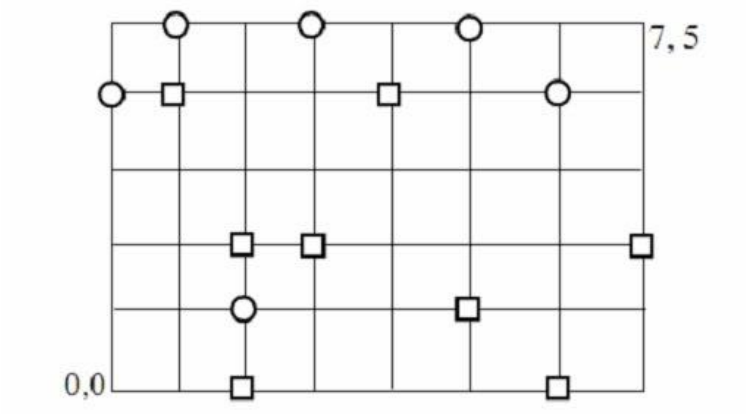
```
void in(thung th[],int n)
{
    for(int i=0;i<n;i++)
    {
        printf("%d %d\n",th[i].luongnuoc,th[i].dungtich);
    }
}

void inthung(thung th[],int n)
{
    for(int i=n-1;i>-1;i--)
    {
        if(th[i].dungtich>th[i].luongnuoc)
        {
            printf("%d %d\n",th[i].luongnuoc,th[i].dungtich);
        }
        else
            break;
    }
}

int main()
{
    thung th[MAX];
    int n;
    if(Nhap(th,n)==0)
    {
        printf("loi doc file");
        return 0;
    }

    Sort(th,n);
    //in(th,n);
    DoNuoc(th,n);
    inthung(th,n);
    //in(th,n);
    return 0;
}
```

BÀI :ĐỊA ĐẠO (Cao Đẳng 2005)

Trong các cuộc kháng chiến chống xâm lược, cha ông ta đã xây dựng các địa đạo rất lớn dưới lòng đất với các tuyến đường giao thông ngầm chằng chịt, vừa bảo đảm an toàn, vừa giữ bí mật tuyệt đối. Trong địa đạo này giao thông đi lại phải tuân thủ các qui định chặt chẽ, tất cả mọi người đều phải di chuyển dọc theo các tuyến đường và thực hiện nghiêm ngặt các chỉ dẫn giao thông trên đường.

Một trong các địa đạo như vậy bao gồm N đường dọc và M đường ngang được mô tả như một lưới ô vuông kích thước N x M. Các đường ngang đánh số từ 0 đến M-1 từ dưới lên trên, các đường dọc đánh số từ 0 đến N-1 từ trái sang phải. Tại một số vị trí giao giữa các đường người ta đặt các biển chỉ dẫn dạng ô vuông hoặc ô tròn với ý nghĩa như sau: khi di chuyển theo các đường tại các nút giao thông, nếu gặp chỉ dẫn ô vuông thì bắt buộc rẽ trái, nếu gặp chỉ dẫn ô tròn thì bắt buộc rẽ phải, còn nếu không có chỉ dẫn thì phải đi thẳng. Sơ đồ sau cho ta một hình ảnh các

đường của địa đạo và các chỉ dẫn.
Bạn có nhiệm vụ dẫn một đoàn khách tham quan đi theo các đường của địa đạo, xuất phát từ vị trí (0,0) và tuân thủ chỉ dẫn tại các nút giao thông. Từ vị trí **ban đầu bạn có thể đi theo một trong hai hướng (ngang hoặc dọc)**. Đường đi của bạn sẽ dừng lại nếu xảy ra một trong hai tình huống sau:
1. Không thể đi tiếp được nữa.
2. Nút giao thông sắp đến theo hành trình là nút giao thông đã đi qua.
Độ dài của đoạn đường đã đi là tổng số các nút giao thông đã đi qua kể cả vị trí xuất phát và vị trí kết thúc.



Yêu cầu: tính độ dài của đoạn đường có thể đi được trong địa đạo.
Dữ liệu: vào từ file văn bản PIPE.INP có dạng sau:
• Dòng đầu tiên chứa 3 số tự nhiên là N, M và K với N, M là kích thước của lưới mô tả địa đạo. N, M < 100. K là số lượng các chỉ dẫn có tại các nút giao thông trong địa đạo. K < 1000. Các số cách nhau bởi dấu cách.
• K dòng tiếp theo mô tả tọa độ và tính chất của các chỉ dẫn tại các nút giao thông tương ứng. Mỗi dòng bao gồm 3 số nguyên dạng X Y Z. Ở đây X, Y là toạ độ của vị trí biển chỉ dẫn ($0 \le X \le N-1$, $0 \le Y \le M-1$), Z nhận giá trị 0 hoặc 1 với ý nghĩa: 0 là chỉ dẫn rẽ phải và 1 là chỉ dẫn rẽ trái. Các số cách nhau bởi dấu cách.
Kết quả: ghi ra file văn bản PIPE.OUT bao gồm một dòng chứa đúng 2 số tự nhiên theo thứ tự tăng dần là độ dài của hai đường đi trong địa đạo xuất phát từ vị trí ban đầu.
Ví dụ:
PIPE.INP
8 6 14
0 4 0
1 4 1
1 5 0
2 0 1
2 1 0
2 2 1
3 2 1
3 5 0
4 4 1
5 1 1

5 5 0
6 0 1
6 4 0
7 2 1

PIPE.OUT
13 19

```
// Code của @hienclubvn
// Ý tưởng : duyệt
#include<stdio.h>
#include<conio.h>
#include<malloc.h>
#define Input "PIPE.INP"
#define Output "PIPE.OUT"
#define MAX 100
int A[MAX][MAX],B[MAX][MAX];
int x1=0,y1=0,x,y;
int count=1;
void Init(int n,int m)
{
    for(int i=0;i<n;i++)
        for(int j=0;j<m;j++)
            A[i][j]=2;
}
void Init_1(int n,int m)
{
    for(int i=0;i<n;i++)
      for(int j=0;j<m;j++)
        if(i==0&&j==0)B[i][j]=1;
        else B[i][j]=0;

}
void nhap(int &n,int &m)
{
    FILE *f=fopen(Input,"r");
    int k;
    fscanf(f,"%d",&n);
    fscanf(f,"%d",&m);
    fscanf(f,"%d",&k);
    Init(n,m); // Creat Arr A
    int temp1,temp2,temp3;
    for(int i=0;i<k;i++)
    {
        fscanf(f,"%d",&temp1);
        fscanf(f,"%d",&temp2);
        fscanf(f,"%d",&temp3);
        A[temp1][temp2]=temp3;
    }
    for(int i=0;i<m;i++)
    {
        for(int j=0;j<n;j++)
            printf("%d ",A[j][i]);
        printf("\n");
    }
```

```
    fclose(f);
}
void dithang(int &x,int &y)
{
    if(y==y1+1) {x1=x;y1=y;y+=1;return;}
    if(y==y1-1) {x1=x;y1=y;y-=1;return;}
    if(x==x1+1) {x1=x;y1=y;x+=1;return;}
    if(x==x1-1) {x1=x;y1=y;x-=1;return;}
}
void queo_phai(int &x,int &y)
{
    if(y==y1+1) {x1=x;y1=y;x+=1;return;}
    if(y==y1-1) {x1=x;y1=y;x-=1;return;}
    if(x==x1+1) {x1=x;y1=y;y-=1;return;}
    if(x==x1-1) {x1=x;y1=y;y+=1;return;}
}
void queo_trai(int &x,int &y)
{
    if(y==y1+1) {x1=x;y1=y;x-=1;return;}
    if(y==y1-1) {x1=x;y1=y;x+=1;return;}
    if(x==x1+1) {x1=x;y1=y;y+=1;return;}
    if(x==x1-1) {x1=x;y1=y;y-=1;return;}
}
void Process(int n,int m)
{
     while(x<n&&y<m&&B[x][y]==0)
     {
        switch(A[x][y])
        {
          case 2: B[x][y]=1;dithang(x,y); count++;break;
          case 1: B[x][y]=1;queo_trai(x,y);count++;break;
          case 0: B[x][y]=1;queo_phai(x,y);count++;break;
        }
     }
}
void Print(int n,int m)
{
    printf("\n");
    for(int i=0;i<m;i++)
    {
       for(int j=0;j<n;j++) printf("%d ",B[j][i]);
       printf("\n");
    }
}
int main()
{
    int m,n;
    nhap(n,m);
    // --------- duong di thu 1:
    Init_1(n,m);
    x=0;y=1;
    Process(n,m);
    Print(n,m);
    printf("%d\n\n",count);
    // ---------- duong di thu 2:
    Init_1(n,m);
    x=1;y=0;
    x1=0;y1=0;
```

```
    count=1;
    Process(n,m);
    Print(n,m);
    printf("%d\n",count);
    getch();
}


// Code của @vietduc
#include<iostream>
#include <fstream>
using namespace std;
struct node
{
    int value;
    bool ed;
};
node **pipe;

int timduong(int h, int i=1, int j=1)
{
    cout << i << "  "<<j<<endl;
    if (pipe[i][j].ed==false)
        return 0;
    else if (pipe[i][j].value==3)
        return -1;
    else if (pipe[i][j].value==1)
    {
        h+=1;
        if (h>3) h-=4;
        pipe[i][j].ed=false;
    }
    else if(pipe[i][j].value==2)
    {
        h-=1;
        if (h<0) h+=4;
        pipe[i][j].ed=false;
    }
    if (h==0)
        i++;
    else if (h==1)
        j++;
    else if (h==2)
        i--;
    else if (h==3)
        j--;
    int y = 1+timduong(h,i,j);
    pipe[i][j].ed=true;
    return y;
}


int main()
{
    ifstream infile("pipe.txt");
    int m,n,k,i,j;
    infile >>n >>m >>k;
    pipe=new node*[m+2];
```

```
    for (i=0; i<m+2; i++)
    {
        pipe[i]=new node[n+2];
        for (j=0; j<n+2; j++)
        {
            if (i==0 || i== m+1 || j==0 || j== n+1)
                pipe[i][j].value=3;
            else pipe[i][j].value=0;
            pipe[i][j].ed=true;
        }
    }
    for (i=0; i<k; i++)
    {
        int a, b, c;
        infile >> a >> b >> c;
        pipe[b+1][a+1].value=c+1;
    }
    int a=timduong(0);
    int b=timduong(1);
    cout << a <<" "<< b << endl;
    return 0;
}
```

```
// Code của @Sounj
#include <fstream>
#include <iostream>
#include <conio.h>
using namespace std;
#define MAX_SIZE 100

struct ban_do
{
    int N, M;
    int D[MAX_SIZE][MAX_SIZE];
};

bool nhap(char* ten_file, ban_do& bd)
{
    ifstream f(ten_file);
    if(!f) return false;

    int K, i, j, x, y;
    f>>bd.N>>bd.M>>K;
    for(i=0; i<bd.N; i++)
        for(j=0; j<bd.M; j++)
            bd.D[i][j] = 1;

    for(i=0; i<K; i++)
    {
        f>>x>>y>>j;
        bd.D[x][y] = j+2;
    }

    return true;
}

int dinh_huong(int huong, int chi_thi)
```

```
{
    switch(chi_thi)
    {
    case 2: // re phai
        huong -= 1;
        if(huong<0) huong = 3;
        break;
    case 3: // trai
        huong += 1;
        if(huong>3) huong = 0;
        break;
    }
    return huong;
}

int di_qua_dia_dao(int x, int y, int huong, ban_do& bd)
{
    const int const lech[4][2]={{1,0},{0,1},{-1,0},{0,-1}};

    int kq=0;
    while(0<=x && x<bd.N && 0<=y && y<bd.M && bd.D[x][y]>0)
    {
        huong = dinh_huong(huong, bd.D[x][y]);
        ++kq;
        bd.D[x][y] = -bd.D[x][y]; // da di qua
        // doi huong
        x += lech[huong][0];
        y += lech[huong][1];
    }

    return kq;
}

void reset_bando(ban_do& bd)
{
    int i,j;
    for(i=0; i<bd.N; i++)
        for(j=0; j<bd.M; j++)
            if(bd.D[i][j] < 0)
                bd.D[i][j] = -bd.D[i][j];
}

bool Xuat(char* ten_file, int kq1, int kq2)
{
    ofstream f(ten_file);
    if(!f) return false;

    int tg;
    if(kq1>kq2)
    {
        tg = kq1; kq1 = kq2; kq2 = tg;
    }
    f<<kq1<<" "<<kq2;

    return true;
}

void main()
```

```
{
    ban_do bd;
    if(nhap("PIPE.INP", bd))
    {
        int kq1, kq2;
        kq1 = di_qua_dia_dao(0, 0, 1, bd);
        reset_bando(bd);
        kq2 = di_qua_dia_dao(0, 0, 0, bd);
        Xuat("PIPE.OUT", kq1, kq2);
        cout<<"xong ... ";
    }else cout<<"Xay ra loi khi doc du lieu tu file !";
    getch();
}
```

```
// Code của @AlexBlack
#include<stdio.h>
void Init(int **&A,int m,int n)
{
    for(int i=0;i<m;i++)
        for(int j=0;j<n;j++)
            A[i][j]=2;
}

void Init(int *A,int n)
{
    for(int i=0;i<n;i++)
        A[i]=0;
}
void nhap(int **&A,int &m,int &n)
{
    FILE *f=fopen("input.txt","rt");
    fscanf(f,"%d",&n);
    fscanf(f,"%d",&m);

    A=new int*[m];
    for(int i=0;i<m;i++)
        A[i]=new int[n];
    Init(A,m,n);
    int temp1,temp2,temp3;
    fscanf(f,"%d",&temp1);
    while(!feof(f))
    {
        fscanf(f,"%d",&temp1);
        fscanf(f,"%d",&temp2);
        fscanf(f,"%d",&temp3);
        A[temp2][temp1]=temp3;
    }
    for(int i=0;i<m;i++)
    {
        for(int j=0;j<n;j++)
            printf("%d ",A[i][j]);
        printf("\n");
    }
    fclose(f);
}
```

```
int run(int **A,int m,int n,int i,int j,int exi,int exj,int *B)
{
    int vt=1;
    while(true)
    {
        if(i>-1&&i<m&&j>-1&&j<n)
        {
            if(B[i*n+j]==1)
            {
                vt--;
                break;
            }
            if(A[i][j]==0)//rẽ phải
            {
                if(j==exj)//quá khứ đi từ dưới lên trên hoặc từ trên xg dưới
                {
                    if(i>exi)//đi từ dưới lên trên
                    {
                    exi=i;
                    j++;
                    }
                    else//đi từ trên xuốn dưới
                    {
                    exi=i;
                    j--;
                    }
                }
                else//quá khư đi từ trái sang phải hoặc từ phải sang trái
                {
                    if(j>exj)//đi từ trái sang phải
                    {
                    exj=j;
                    i--;
                    }
                    else//đi từ phải sang trái
                    {
                        exj=j;
                        i++;
                    }
                }
                vt++;
            }
            else
            if(A[i][j]==1)//rẽ trái
            {
                if(j==exj)//quá khứ đi từ dưới lên trên hoặc từ trên xg dưới
                {
                    if(i>exi)//đi từ dưới lên
                    {
                    exi=i;
                    j--;
                    }
                    else//đi từ trên xuống dưới
                    {
                        exi=i;
                    j++;
                    }
                }
```

```
                else//quá khư đi từ trái sang phải hoặc từ phải sang trái
                {
                    if(j>exj)//đi từ trái sang phải
                    {
                    exj=j;
                    i++;
                    }
                    else//đi từ phải sang trái
                    {
                        exj=j;
                    i++;
                    }
                }
                vt++;
            }
            else//đi thẳng
            {
                if(j==exj)//quá khứ đi từ dưới lên trên hoặc từ trên xg dưới
                {

                if(i>exi)//đi từ dưới lên
                {exi=i;
                    i++;
                }
                else//đi từ trên xuống
                {
                    exi=i;
                    i--;
                }
                }
                else//quá khư đi từ trái sang phải hoặc từ phải sang trái
                {

                    if(j>exj)//đi từ trái sang phải
                    {exj=j;
                        j++;
                    }
                    else//đi từ phải sang trái
                    {exj=j;
                        j--;
                    }
                }
                vt++;
            }
            B[exi*n+exj]=1;
        }
        else
            break;
    }
    return vt;
}

void duongdi(int **A,int m,int n)
{
    int *B;
    int sovitri;
    B=new int[n*m];
    Init(B,n*m);
```

```
    sovitri=run(A,m,n,1,0,0,0,B);
    printf("theo chieu doc: %d\n",sovitri);
    Init(B,n*m);
    sovitri=run(A,m,n,0,1,0,0,B);
    printf("theo chieu ngang: %d",sovitri);
}

int main()
{
    int **A;
    int m,n;

    nhap(A,m,n);
    duongdi(A,m,n);
    return 0;
}
```

Bài: Tìm đặc trưng ảnh (Ko chuyên 2005)

Một nhóm nghiên cứu xử lý ảnh của trường ĐH Khoa học tự nhiên, ĐHQG Tp.HCM đang giải quyết bài toán nhận dạng mặt người trong ảnh. Ảnh chụp mặt người sau khi đã xử lý là một bảng vuông A kích thước N x N ($10 \le N \le 800$) với mỗi ô (I,J) ($1 \le I, J \le N$) có giá trị từ 0 đến 255 là mức xám của ảnh tại ô này (trong đó 0 là màu nền). Để xác định vị trí có thể là mặt người, nhóm cần thống kê các đặc trưng có dạng hình vuông kích thước K x K ($1 \le K \le 40$) trong đó tất các các giá trị trong hình vuông đều phải khác 0.

Yêu cầu : Từ một ảnh chụp mặt người, hãy giúp nhóm nghiên cứu đếm tất cả các đặc trưng có trong ảnh đó.

Dữ liệu : Vào từ file văn bản FEATURE.INP trong đó :

• Dòng đầu chứa hai sô N và K

• Dòng thứ I trong N dòng tiếp theo chứa tương ứng dòng thứ I của bảng A.

Các số ghi trên một dòng được ghi cách nhau bởi ít nhất một khoảng trắng. Mỗi dòng có N số nguyên. Dòng thứ I là các giá trị của N phần tử trong dòng thứ I-1 trong bảng vuông A

Kết quả : Ghi ra file văn bản FEATURE.OUT số lượng đặc trưng tìm được.

Ví dụ:

FEATURE.INP

```
6 2
0 12 15 0 33 30
17 19 23 15 16 0
11 12 0 14 14 0
0 10 11 8 10 0
0 8 7 12 0 0
0 0 11 13 0 0
```

FEATURE.OUT

```
7
```

```
/* Code của @AlexBlack
      Ý tưởng: duyệt toàn bộ ma trận, kiểm tra những vùng có diện
      tích KxK xem có phải là mặt người hay không. nếu phải thì
      tăn biến đếm lên.
*/
#include<stdio.h>
#define input "FEATURE.INP"
typedef struct Anh
{
    int n;
    int k;
    int **Array;
};

void Init(Anh &a)
{
    for(int i=0;i<a.n;i++)
        for(int j=0;j<a.n;j++)
            a.Array[i][j]=0;
}
int Nhap(Anh &a)
{
    FILE *f=fopen(input,"rt");
    if(f==NULL)
        return 0;
    fscanf(f,"%d",&a.n);
    fscanf(f,"%d",&a.k);
    a.n;
    a.Array=new int*[a.n];
    for(int i=0;i<a.n;i++)
    {
        a.Array[i]=new int[a.n];
    }
    //Init(a);
    for(int i=0;i<a.n;i++)
        for(int j=0;j<a.n;j++)
        {
            fscanf(f,"%d",&a.Array[i][j]);
        }
    fclose(f);
    return 1;
}
int KtMatNguoi(Anh a,int p,int q)
{
    for(int i=p;i<p+a.k;i++)
        for(int j=q;j<q+a.k;j++)
        {
            if(a.Array[i][j]==0)
                return 0;
        }
        return 1;
}
int SoMatNguoi(Anh a)
{
```

```
    int sum=0;
    for(int i=0;i<a.n-a.k+1;i++)
        for(int j=0;j<a.n-a.k+1;j++)
        {
            if(KtMatNguoi(a,i,j))
                sum++;
        }
    return sum;
}

//hủy vùng nhớ cấp phát
void Detroy(Anh &r)
{
    for(int i=0;i<r.n;i++)
        delete []r.Array[i];
    delete []r.Array;
}

int main()
{
    Anh a;
    if(!Nhap(a))
    {
        printf("Khong mo duoc file.");
        return 0;
    }
    printf("So mat nguoi trong anh: %d",SoMatNguoi(a));
    Detroy(a);
    return 0;
}



/* Code của @hienclubvn
      giá trị ảnh từ : 0-->255
      kiểm tra KxK (cac gia tri !=0) la OK
*/
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#define IN "FEATURE.INP"
#define OUT "FEATURE.OUT"
int **A,n,k,count=0;
FILE *fin,*fout;
void Read()
{
     fin=fopen(IN,"r");
     fscanf(fin,"%d %d",&n,&k);
     // cap phat bo nho dong
     A=(int**)malloc(n*sizeof(int));
     for(int i=0;i<n;i++)
     {
          A[i]=(int*)malloc(n*sizeof(int));
          //Load File
          for(int j=0;j<n;j++) fscanf(fin,"%d",&A[i][j]);
     }
     fclose(fin);//Close File
}
```

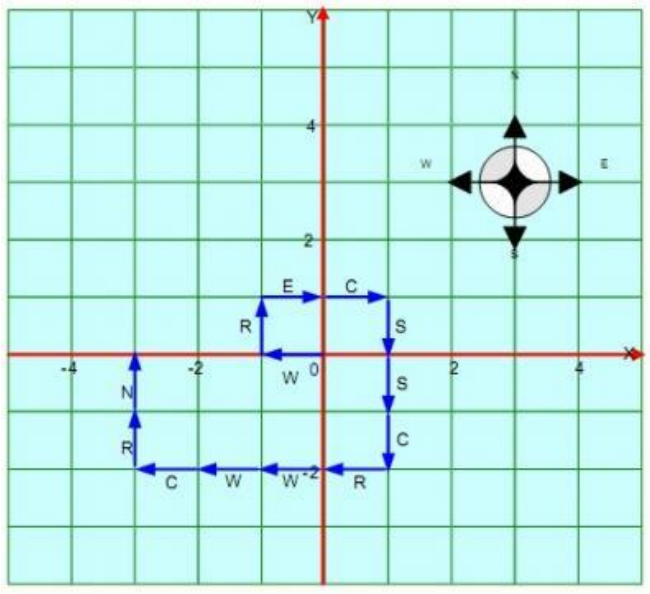
```
int Check(int p,int q)
{
    int i,j;
    for(i=0;i<k;i++)
       for(j=0;j<k;j++)
         if (A[p+i][q+j]==0) return 0;
    return 1;
}
int Process()
{
    int i,j;
    for(i=0;i<=n-k;i++)//dong
       for(j=0;j<=n-k;j++)//cot
          if(Check(i,j)) count++;
}
void Write()
{
     fout=fopen(OUT,"w");
     fprintf(fin,"%d",count);
     fclose(fout);
}
int main()
{
    Read();
    Process();
    Write();
    return 0;
}
```

```
/* Code của @vietduc
     dịch từng ô một rồi check xem nó có thỏa mãn hay  không,
nếu có thì ++
*/
#include<iostream>
#include<fstream>
using namespace std;
bool check(int **s, int k, int m, int n)
{
    for (int i=0; i<k; i++)
        for (int j=0; j<k; j++)
            if (!s[m+i][n+j])
                return false;
    return true;
}

int main()
{
    int n, k, **s, i, j, t=0;
    ifstream infile("feature.txt");
    infile >>n >> k;
    s=new int*[n];
    for (i=0; i<n; i++)
    {
        s[i]=new int[n];
        for (j=0; j<n; j++)
```

```
            infile >> s[i][j];
    }
    for (i=0; i<=n-k; i++)
        for (j=0; j<n-k; j++)
            if (check(s,k,i,j))
                t++;
    cout << t;
    return 0;
}
```

Bài: Thám hiểm (Đề Khối Ko Chuyên 2005)

Đoàn thám hiểm Nam cực xuất phát từ Trạm nghiên cứu đặt ở điểm có toạ độ (0, 0). Trục OX chạy từ tây sang đông, trục OY chạy từ nam lên bắc. Ban đầu Đoàn đi theo một trong số 4 hướng **Đông (E), Tây (W), Nam (S) hoặc Bắc (N)**. Sau khi đi một quảng đường là một đơn vị độ dài Đoàn dừng lại thực hiện các đo đạc khảo sát cần thiết và tuỳ theo tình hình thực tế,trưởng đoàn sẽ quyết định đi tiếp theo hướng nào. Để đánh dấu các điểm trên bản đồ khai thác và dễ dàng xác định đường trở về lịch trình di chuyển trong nhật ký được ghi dưới dạng xâu các ký tự trong tập {E, W, S, N}. Tuy vậy thời tiết ở địa cực nổi tiếng là đỏng đảnh. Bão tuyết có thể nổi lên bất kỳ lúc nào kèm theo các cơn bão từ làm rối loạn la bàn. Khi đó đường đi sẽ được ghi nhận bằng các ký tự **L - Rẽ trái**, **R - Rẽ phải**, **B – Đi ngược lại hướng vừa đi** hoặc **C - Tiếp tục đi thẳng theo hướng cũ**. Mỗi khi bão từ chấm dứt, la bàn hoạt động bình thường trở lại lịch trình di chuyển lại được ghi nhận bằng các ký tự E, W, S, N. Ở thời điểm ban đầu của cuộc thám hiểm thời tiết tốt, không có bão từ. Trong suốt cuộc hành trình có thể không có trận bão từ nào hoặc bão từ có thể xảy ra nhiều lần. Ví dụ, lịch trình ở hình bên có thể được ghi nhận bằng xâu WRECSSCRWWCRN



Yêu cầu: Cho một lịch trình di chuyển xác định bởi xâu Z độ dài không quá 200 gồm các ký tự trong tập **{E, W, S, N, L ,R, B, C}**, bắt đầu bằng một trong số 4 ký tự đầu tiên. Hãy tính khoảng cách theo đường chim bay từ điểm dừng cuối của đoàn thám hiểm tới Trạm nghiên

cứu mà Đoàn xuất phát.
Khoảng cách d theo đường chim bay giữa hai điểm có toạ độ (X1, Y1) và (X2,Y2) được tính theo công thức

$$d = \sqrt{(X_1 - X_2)^2 + (Y_1 - Y_2)^2}\ .$$

Dữ liệu: Vào từ file văn bản EXPLORE.INP gồm một dòng chứa xâu Z xác định một lịch trình di chuyển của đoàn thám hiểm.

Kết quả: Đưa ra file văn bản EXPLORE.OUT một số thực có 3 chữ số sau dấu chấm thập phân, đó là khoảng cách tìm được với dữ liệu vào đã cho.

Ví dụ 1:
EXPLORE.INP
WNRN

EXPLORE.OUT
2.000

Ví dụ 2:
EXPLORE.INP
WRECSSCRWWCRN

EXPLORE.OUT
3.000

```
/* Code của @AlexBlack
ta chỉ cần phải tìm tọa độ điểm cuối thôi.
ý tưởng là dựa trên 4 nguyên lý cơ bản về hướng đông tây nam
bắc. rồi rẽ thì dựa vào 4 hướng đó mà tính toán.
lưu vị trí cũ để xét sự tương quan của vị trí mới với vị trí cũ
mà tính toán cho chính xác
*/
#include "stdio.h"
#include "math.h"
void East(int &x,int &y,int &ex,int &ey)
{
    ex=x,ey=y,x++;
}

void West(int &x,int &y,int &ex,int &ey)
{
    ex=x,ey=y,x--;
}

void North(int &x,int &y,int &ex,int &ey)
{
```

```
    ex=x,ey=y,y++;
}

void South(int &x,int &y,int &ex,int &ey)
{
    ex=x,ey=y,y--;
}

void Right(int &x,int &y,int &ex,int &ey)
{
    switch(x-ex)
    {
    case 1:South( x, y, ex, ey);break;
    case -1:North( x, y, ex, ey);break;
    case 0:
        switch(y-ey)
        {
        case 1:East( x, y, ex, ey);break;
        case -1:West( x, y, ex, ey);break;
        };break;
    }
}

void Left(int &x,int &y,int &ex,int &ey)
{
    switch(x-ex)
    {
    case 1:North( x, y, ex, ey);break;
    case -1:South( x, y, ex, ey);break;
    case 0:
        switch(y-ey)
        {
        case 1:West( x, y, ex, ey);break;
        case -1:East( x, y, ex, ey);break;
        };break;
    }
}

void Continue(int &x,int &y,int &ex,int &ey)
{

    switch(x-ex)
    {
    case 1:East( x, y, ex, ey);break;
    case -1:West( x, y, ex, ey);break;

    case 0:
        switch(y-ey)
        {
        case 1:North( x, y, ex, ey);break;
        case -1:South( x, y, ex, ey);break;
        };break;
    }
}

void Back(int &x,int &y,int &ex,int &ey)
{
    switch(x-ex)
```

```
    {        case 1:West( x, y, ex, ey);break;
    case -1:East( x, y, ex, ey);break;

    case 0:
        switch(y-ey)
        {
        case 1:South( x, y, ex, ey);break;
        case -1:North( x, y, ex, ey);break;
        };break;
    }
}



void xl(int &x,int &y,int &ex,int &ey,char c)
{
    switch(c)
    {
    case 'E':East( x, y, ex, ey);break;
    case 'W':West( x, y, ex, ey);break;
    case 'N':North( x, y, ex, ey);break;
    case 'S':South( x, y, ex, ey);break;
    case 'L':Left( x, y, ex, ey);break;
    case 'R':Right( x, y, ex, ey);break;
    case 'C':Continue( x, y, ex, ey);break;
    case 'B':Back( x, y, ex, ey);break;
    }
}

double khoangcach(char *A,int n)
{
    int x=0,y=0,ex=0,ey=0;
    for(int i=0;i<n;i++)
    {
        xl(x,y,ex,ey,A[i]);
    }

    return sqrt((double)(x*x+y*y));
}


int main()
{
    char A[]={'W','R','E','C','S','S','C','R','W','W','C','R','N'};int n=13;

    double kq=khoangcach(A,n);
    printf("%.1lf",kq);
}
```

Bài : Radar (Đề Ko chuyên 2006)

Một vùng biển hình chữ nhật được chia lô thành m hàng được đánh số từ 1 đến m từ trên xuống dưới và n cột được đánh số từ 1 đến n từ trái sang phải. Lô nằm ở vị trí giao của hàng p

(1≤ p ≤m) và cột q (1≤ q ≤n) được gọi là lô có tọa độ (p, q). Để bảo vệ các giàn khoan dầu trên vùng biển này người ta bố trí một số radar tại một số lô. Mỗi radar có khả năng phát hiện tầu thuyền tại chính lô đó và 8 lô lân cận (4 lô chung cạnh và 4 lô chung đỉnh) kể cả trên biên của các lô này. Một lô trên vùng biển được coi là an toàn nếu tàu từ ngoài vùng biển trên muốn vào trong lô đó thì dù đi theo đường đi như thế nào cũng đều bị ít nhất một radar phát hiện.
Yêu cầu: Cho kích thước của vùng biển và vị trí của các lô được bố trí radar. Hãy xác định tổng số lô an toàn nằm trong vùng biển này.
Dữ liệu: Vào từ tệp văn bản RADAR.INP có định dạng như sau:
• Dòng đầu ghi hai số nguyên dương m và n (1≤ m, n ≤300) là kích thước (hàng và cột) của vùng biển. Hai số được ghi cách nhau một dấu cách.
• Dòng thứ hai ghi số nguyên k (1 ≤ k ≤ m x n) là số các radar được bố trí.
• Trên dòng thứ i trong k dòng tiếp theo ghi hai số nguyên dương p, q (1 ≤ p ≤ m, 1≤ q ≤ n) là tọa độ lô bố trí radar thứ i. Hai số được ghi cách nhau một dấu cách.
Kết quả: Ghi ra tệp văn bản RADAR.OUT một số nguyên dương là tổng số các lô an toàn trong vùng biển.
Ví dụ:
RADAR.INP
8 8
4
1 1
2 4
4 1
4 3

RADAR.OUT
23

```
/* Code của @pannaturo
   -  Ý tưởng:
      Giả sử có vùng n x m và tọa độ các rada
      Ta lưu nó vào trong mảng kích thước ( n + 2 ) * ( m + 2 )
      Ví dụ kích thước 3 x 3 và rada có tọa độ ( 1, 1 )
      như thế bình thường vị trí rada là ( 0, 0 ) trong mảng 3 x
      3
      Bây giờ ta lưu nó vào mảng 5 x 5 => tọa độ ( 1 x 1 )
      Từ đó khi xét các ô an toàn hay không ta tránh phải xét xem
      rada có nằm trên biên hay không.
      Cách này sẽ tốn chi phí hơn bình thường.
*/
#include <iostream>
#include <fstream>
using namespace std;
#define MAX 50
int main()
{
    int nCol, nRow, nRada;
    fstream fin( "input.txt", ios::in );
    fin >> nCol >> nRow >> nRada;
```

```
    bool arr[ MAX ][ MAX ];
    for( int i = 0; i <= nRow + 1; i++ )
        for( int j = 0; j <= nCol + 1; j++ )
            arr[ i ][ j ] = false;
    int xx, yy;
    for( int i = 0; i < nRada; i++ )
    {
        fin >> xx >> yy;
        arr[xx][yy]=arr[xx][yy -1]=arr[xx][ yy + 1]=true;
        arr[xx-1][yy-1]=arr[xx-1][yy]=arr[xx -1][yy+1]=true;
        arr[xx +1][yy-1] = arr[xx+1][yy] = arr[xx+1][yy+1]=true;
    }
    fin.close();
    int count = 0;
    for( int i = 1; i <= nRow; i++ )
        for( int j = 1; j <= nCol; j++ )
            if ( arr[ i ][ j ] )
                count++;
    fstream fout( "output.txt", ios::out );
    fout << count;
    fout.close();
    return 0;
}
```

```
/*    Code của @hienclubvn
      Ý tưởng: Cái nào an toàn cho bằng 1
*/
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#define INPUT "RADAR.INP"
#define OUTPUT "RADAR.OUT"
int **B,n,m;
int dx[3]={-1,0,1};
int dy[3]={-1,0,1};
void Rada(int x,int y)
{
     int i,j,u,v;
    for(i=0;i<3;i++)
    {
        u=x+dx[i];
        for(j=0;j<3;j++)
        {
           v=y+dy[j];
           if((u>=1&&u<=n)&&(v>=1&&v<=m))
           {
              B[u-1][v-1]=1;
           }
        }
    }
}
void Print()
{
     int i,j;
     for(i=0;i<m;i++)
```

```
      {
          for(j=0;j<m;j++)
          printf("%d  ",B[i][j]);
          printf("\n");
      }
}
int SumSafe() // Tổng các ô an toàn
{
    int i,j,temp=0;
    for(i=0;i<m;i++)
      for(j=0;j<n;j++)
      if(B[i][j]==1)temp++;
    return temp;
}
int main()
{
    //Read File
    FILE *fin;
    fin=fopen(INPUT,"r");
    // Cap phat bo nho
    fscanf(fin,"%d %d\n",&m,&n);
    B=(int**)malloc(m*sizeof(int));
    for(int i=0;i<m;i++)
    {
        B[i]=(int*)malloc(n*sizeof(int));
        for(int j=0;j<n;j++) B[i][j]=0;
    }
    int temp,x,y;
    fscanf(fin,"%d\n",&temp);
    for(int i=0;i<temp;i++)
    {
        fscanf(fin,"%d %d\n",&x,&y);
        Rada(x,y);
    }
    fclose(fin);
    //Xuat ra man hinh
    Print();
    printf("Tong Cell Safe = %d",SumSafe());
    //Ghi vao File
    FILE *fout;
    fout=fopen(OUTPUT,"w");
    fprintf(fout,"%d",SumSafe());
    fclose(fout);
    getch();
}


/* Code của @AlexBlack
     Ý tưởng : xét những chỗ không phải là rada kiểm tra xem chỗ
     đó có phải là vùng an toàn hay không
     ở đây mình sử dụng mảng từ 1-m,1-n để bỏ qua việc kiểm tra
     điều kiện biên.
*/
#include<stdio.h>
#define input "water.txt"
typedef struct Rada
{
```

```
    int n,m;
    int **Array;
    int nrada;
};

void Init(Rada &r)
{
    for(int i=0;i<=r.m;i++)
        for(int j=0;j<=r.n;j++)
            r.Array[i][j]=0;
}
int Nhap(Rada &r)
{
    FILE *f=fopen(input,"rt");
    if(f==NULL)
        return 0;
    fscanf(f,"%d",&r.m);
    fscanf(f,"%d",&r.n);
    r.m++;
    r.n++;
    r.Array=new int*[r.m+1];
    for(int i=0;i<=r.n;i++)
    {
        r.Array[i]=new int[r.n+1];
    }
    Init(r);
    fscanf(f,"%d",&r.nrada);
    int p,q;
    while(!feof(f))
    {
        fscanf(f,"%d",&p);
        fscanf(f,"%d",&q);
        r.Array[p+1][q+1]=1;
    }
    fclose(f);
    return 1;
}

void ChanHoa(Rada &r)
{
    for(int i=1;i<r.m;i++)
        for(int j=1;j<r.n;j++)
        {
            if(r.Array[i][j]==0)
            {
                if(r.Array[i-1][j-1]==1||r.Array[i-1][j]==1||r.Array[i-
1][j+1]==1)r.Array[i][j]=2;
                if(r.Array[i][j-1]==1||r.Array[i][j+1]==1)r.Array[i][j]=2;
                if(r.Array[i+1][j-
1]==1||r.Array[i+1][j]==1||r.Array[i+1][j+1]==1)r.Array[i][j]=2;

            }
        }
}

void Xuat(Rada r)
{
    for(int i=1;i<r.m;i++)
```

```
    {
        for(int j=1;j<r.n;j++)
            printf("%d ",r.Array[i][j]);
        printf("\n");
    }
}

void XuatAnToan(Rada r)
{
    ChanHoa(r);
    int sum=0;
    for(int i=1;i<r.m;i++)
        for(int j=1;j<r.n;j++)
        if(r.Array[i][j]!=0)
            sum+=1;
    printf("Tong so o an toan: %d",sum);
}

//hủy vùng nhớ cấp phát
void Detroy(Rada &r)
{
    for(int i=0;i<r.m+1;i++)
        delete []r.Array[i];
    delete []r.Array;
}


int main()
{
    Rada r;
    if(!Nhap(r))
    {
        printf("Khong mo duoc file.");
        return 0;
    }
    XuatAnToan(r);
    Detroy(r);
    return 0;
}
```

Bài: SIÊU MÃ(Đề thi CAO ĐẲNG OLP 2006)

Siêu mã là một loại mã có nhiều ứng dụng quan trọng trong lĩnh vực mã hóa và truyền tin. Trong bài này, ta xét bài toán đơn giản sau đây về siêu mã. Cho u và v là hai xâu kí tự khác rỗng có độ dài hữu hạn. Xâu u được gọi là xâu con của xâu v nếu u có thể nhận được từ v bằng cách xóa bớt ít nhất một kí tự trong v. Một tập X các xâu khác rỗng có độ dài hữu hạn được gọi là siêu mã nếu mọi cặp u, v bất kỳ thuộc X, u không là xâu con của v và v không là xâu con của u.
Cho trước một tập X = {x1, x2, ..., xN} gồm N xâu khác rỗng, mỗi kí tự trong xâu là 0 hoặc 1. Hãy kiểm tra xem X có là một siêu mã hay không?
Dữ liệu: vào từ file văn bản HCODE.INP có định dạng như sau:
• Dòng đầu tiên chứa số nguyên dương N (N ≤ 500);
• Dòng thứ i trong N dòng tiếp theo ghi xâu xi của tập X, độ dài của xâu xi không quá 15, với i = 1, 2, ..., N.

Kết quả: ghi ra file văn bản HCODE.OUT có định dạng như sau:
• Nếu X là siêu mã thì ghi số 1;
• Nếu X không là siêu mã thì dòng đầu tiên ghi số 0, dòng thứ hai ghi chỉ số i nhỏ nhất mà hoặc xi là xâu con của xj hoặc xj là xâu con của xi, với xi, xj thuộc X, $1 \le i < j \le N$.

Ví dụ:

HCODE.INP
5
1111
100101
01011
000
0001000

HCODE.OUT
0
2

HCODE.INP
3
010
1000
11

HCODE.OUT
1

```
// Code của @vietduc
#include <iostream>
#include <fstream>
using namespace std;

bool achuab(char *a, char *b)
{
    while (*a)
    {
        if (*a == *b) a++,b++;
        else a++;
    }
    return (!*b);
}
int main()
{
    ifstream infile("hcode.txt");
    char **x;
    int N;
    infile >> N;
    x=new char*[N];
    for (int i=0; i<N; i++)
    {
        x[i]=new char[20];
```

```
        infile >> x[i];
    }
    for (i=0; i<N; i++)
        for (int j=0; j<N; j++)
            if (i!=j && achuab(x[i],x[j]))
            {
                cout << "0\n"<<i+1<<" "<<j+1<<endl;
                return 0;
            }
    cout << "1\n";
    return 0;
}
```

Bài: Tính điểm (Tập thể ko chuyên 2006)

Trong kỳ thi vấn đáp học sinh phải trả lời các câu hỏi của thầy giáo. Nếu trả lời đúng, thầy giáo đánh dấu bằng ký tự 'C' (Correct), nếu sai thì đánh dấu 'N' (No Correct). Khi học sinh trả lời đúng, thầy sẽ đưa ra câu hỏi tiếp theo khó hơn câu trước, còn khi trả lời sai thầy sẽ cho câu hỏi mới dễ hơn. Sau khi thi xong, kết quả của mỗi học sinh là một xâu các ký tự 'C' và 'N'. Điểm số của học sinh sẽ được tính như sau: Với các câu trả lời sai học sinh không được điểm, với mỗi câu trả lời đúng học sinh nhận được điểm bằng số lần trả lời đúng liên tiếp từ câu trả lời này trở về trước. Ví dụ, nếu kết quả là 'CCNNCNNCCC', thì điểm số sẽ là 1+2+0+1+0+0+1+2+3 = 10.

Yêu cầu: Cho xâu kết quả độ dài không quá 1000, hãy tính điểm của học sinh.

Dữ liệu: Vào từ file văn bản SCORE.INP chứa một xâu kết quả thi.

Kết quả: Đưa ra file văn bản SCORE.OUT điểm số của kết quả thi.

Ví dụ:

SCORE.INP
CCNNCNNCCC

SCORE.OUT
10

```
//Code của @vietduc
#include<iostream>
#include<fstream>
using namespace std;
int main()
{
    ifstream infile("score.txt");
```

```
    char c;
    int b=0,t=0;
    infile.get(c);
    do
    {
        if (c=='C' || c== 'c')
            b++;
        else b=0;
        t+=b;
        infile.get(c);
    }
    while (!infile.eof());
    cout << t;
    return 0;
}



*/   Code của @hienclubvn
     C: đúng, N: ko đúng
*/
#include<stdio.h>
#define IN "SCORE.INP"
#define OUT "SCORE.OUT"
int main ()
{
    FILE *fin,*fout;
    fin=fopen(IN,"r");
    char c;
    int count=0,Sum=0;
    while (!feof(fin))
    {
          c=fgetc(fin);
          if(c=='C')
              count++;
          else count=0;
          Sum+=count;
    }
    fout=fopen(OUT,"w");
    fprintf(fout,"%d",Sum);
    fclose(fin);
    fclose(fout);
}
```

Bài: Phân phòng ở (Tập thể 2006)

Một nhóm N nhà tỷ phú tổ chức đi đánh golf. Tại địa điểm đánh golf có một dãy các ngôi nhà nghỉ nằm trên một địa thế sông núi rất hùng vĩ, có ngôi nhà thì cạnh sông, có ngôi nhà thì cạnh núi ... Mỗi ngôi nhà chỉ ở được một người. Đây cũng chính là lý do khiến các nhà tỷ phú không sao thỏa thuận được người nào sẽ ở ngôi nhà nào. Để giải quyết bế tắc và chiều lòng các tỷ phú, giám đốc khu nghỉ mát quyết định sử dụng M ngôi nhà liền nhau, đánh số từ 1 đến M, để các nhà

tỷ phú lấy ra N phòng trong đó. Nhà tỷ phú thứ i trong nhóm sẽ sử dụng số chứng minh thư Si của mình (không có hai nhà tỷ phú nào có cùng số chứng minh thư) để chọn ra được ngôi nhà mình sẽ ở. Thao tác chọn sẽ như sau:
• Nhà tỷ phú đó sẽ bắt đầu đếm từ ngôi nhà đánh số 1,
• Dừng lại ở ngôi nhà tương ứng với số chứng minh thư của mình,
• Nếu đếm đến ngôi nhà đánh số M thì lại tiếp tục đếm từ ngôi nhà đánh số 1.
Yêu cầu: Hãy giúp giám đốc khu nghỉ mát tìm ra số M bé nhất để không có hai nhà tỷ phú nào chọn cùng một ngôi nhà theo các thao tác vừa nêu ở trên.
Dữ liệu: Vào từ file văn bản ROOM.INP theo qui cách như sau:
• Dòng thứ nhất ghi số nguyên dương N (1<= N <=300) là số các nhà tỷ phú đi đánh golf.
• Dòng thứ hai ghi N số Si (1<= Si <= 1000000) (i=1..N) cách nhau bởi dấu cách, tương ứng là số chứng minh thư của N nhà tỷ phú.
Kết quả: Ghi ra file văn bản ROOM.OUT một số nguyên M, là số lượng phòng ít nhất giám đốc khu nghỉ mát phải sử dụng ứng với dữ liệu vào đã cho.

Ví dụ:
ROOM.INP
2
4 6

ROOM.OUT
3

```
/*    Code của @vietduc
      Ý tưởng: nếu tất cả số CMT đều khác nhau thì m sẽ chạy từ   n-> số CMT lớn nhất, cứ cho m tăng dần lên và mỗi ần như vậy ta sẽ xử nó, nếu nó là số cần tìm thì tất cả các phần dư của số CMT chia cho m đề khác nhau, chỉ cần dựa vào đk này nếu m thỏa mãn thì in ra
*/
#include<iostream>
#include<fstream>
using namespace std;

bool check(int *b, int d, int i)
{
    for (int y=0; y<i; y++)
        if (d==b[y])
            return 0;
    return 1;
}
int main()
{
    ifstream infile("room.txt");
    int i, n, *a, m, M;
    infile >> n;
    a=new int[n];
    for (i=0; i<n; i++)
        infile >> a[i];
    m=n;
    while (m)
```

```
    {
        int *b;
        b=new int[n];
        for (i=0; i<n; i++)
            if (check(b,a[i]%m,i))
            {
                b[i]=a[i]%m;
                if (i==n-1)
                {
                    for (int y=0; y<n; y++)
                        cout << b[y]+1 <<" "; // in ra cac phong duoc o
                    M=m;
                    m=-1;
                }
            }
            else
                break;
        delete []b;
        m++;
    }
    cout <<endl<< M << endl;
    infile.close();
    return 0;
}
```

```
/* Code của @hienclubvn
Giống như giải thuật của @vietduc đã đưa + ý tưởng của @Tadius
Bài toán đưa về dạng đơn giản hơn cho phát biểu sau:
Có một dãy số nguyên dương có n phần tử.
Tìm ra số m nhỏ nhất sao cho mọi phần tử i,j bất kỳ thuộc dãy đã cho thỏa
A[i]%m != A[j]%=m;
Và m >= n. Thì bài toán mới có lời giải
*/
#include<stdio.h>
#include<stdlib.h>
#define IN "ROOM.INP"
#define OUT "ROOM.OUT"
int *A,*B,n;
FILE *fin,*fout;
void ReadFile()
{
     fin=fopen(IN,"r");
     fscanf(fin,"%d",&n);
     // Cap phat bo nho dong
     A=(int*)malloc(n*sizeof(int));
     // Load Mang
     int i;
     for(i=0;i<n;i++) fscanf(fin,"%d",&A[i]);
     fclose(fin);
}
int Check(int j)
{
   for(int i=0;i<j;i++)
     if(B[j]==B[i]) return 0;
```

```
    return 1;
}
void Write(int tmp)
{
    fout=fopen(OUT,"w");
    fprintf(fout,"%d",tmp);
    fclose(fout);
}
int main()
{
     ReadFile();
     int i,m=n,Flag=0;
     // Cho m tang tu n++, den khi thoa man
     do
     {
          int temp=m;
          // Creat Arr phu
          B=(int*)malloc(n*sizeof(int));
          for(i=0;i<n;i++)
          {
               B[i]=A[i]%m;
               if(i && !Check(i))
               {
                    m++;
                    free(B);
                    break;
               }
          }
          if (temp==m) Flag=1;

     }while(!Flag);
     free(A);
     Write(m);
}
```

Bài: Dãy Số (Ko chuyên 2008)

Cho dãy số A0,A1,A2,…..
Trong đó: A0=0;
A_i là số nguyên dương nhỏ nhất lớn hơn A_{i-1} và trong biểu diễn thập phân của A_i ko có chứa chữ số trong biểu diễn thập phân của A_{i-1} với i>=1
Như vậy các phần tử đầu tiên của dãy A là:

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
a	0	1	2	3	4	5	6	7	8	9	10	22	30	41	50

Yêu cầu: cho số tự nhiên N Hãy tìm A_N

Dữ liệu : từ File **NUMSEQ.INP** trong đó chứa duy nhất số N (0=<N<=500)
Kết quả: ghi ra File **NUMSEQ.OUT** giá trị AN
Ví dụ:

NUMSEQ.INP
0
NUMSEQ.INP
12
NUMSEQ.INP
27

NUMSEQ.OUT
0
NUMSEQ.OUT
30
NUMSEQ.OUT
911

```
/* Code của @hiencubvn
Giai thuat : dua vao quy luat de CODE
0 1 2 3 4 5 6 7 8 9 10 22 30 41 50 61 70 81 90 111
200 311 400 511 600 711 800 911   == 8 so (3 chu so)
2000 3111 4000 5111 6000 7111 8000 9111    == 8 so  (4 chu so)
*/
#include<stdio.h>
#include<conio.h>
#define INPUT "NUMSEQ.INP"
#define OUTPUT "NUMSEQ.OUT"
int main()
{
    int n, A[20]={0,1,2,3,4,5,6,7,8,9,10,22,31,40,50,61,70,81,90,111}; // 0-
>19
    FILE *fin,*fout;
    fin=fopen(INPUT,"r");
    fscanf(fin,"%d",&n);
    fclose(fin);
    fout=fopen(OUTPUT,"w");
    //Process
    int heso,chiso,i;
    if(n<=19)
    {
       fprintf(fout,"%d",A[n]);
       fclose(fout);
    }
    else
    {
        n-=20;
        chiso=n/8+2;// chi so chay
        heso=n%8+2; // he so dau
        if(n%2==0)
        {
             fprintf(fout,"%d",heso);
             for(i=0;i<chiso;i++) fprintf(fout,"%d",0);
             fclose(fout);
        }
        else
        {
            fprintf(fout,"%d",heso);
            for(i=0;i<chiso;i++)fprintf(fout,"%d",1);
```

```
            fclose(fout);
        }
    }
    return 0;
}
```

Bài: Đào tạo từ xa (Olympic tin học SV 2009)

Một trong những vấn đề phải giải quyết trong đào tạo từ xa là tìm hiểu xem học viên có thực sự ngồi trước màn hình theo dõi các bài giảng bắt buộc hay không. Học viên có thể tải các bài giảng về và dùng các phần mềm chuyên dụng do Trung tâm đào tạo cung cấp để xem và học vào bất cứ thời điểm nào thuận tiện đối với mình. Vì vậy, cần có những phương pháp kiểm tra tế nhị và khoa học để biết chính xác tình hình học tập của học viên.
Thực tế cho thấy rằng, nếu thông tin hiện lên trên màn hình với khoảng thời gian ít hơn 1/24 giây thì mắt người không kịp ghi nhận được hình ảnh nhưng não bộ vẫn tiếp nhận thông tin! Dựa vào tính chất này phần mềm giảng dạy thiết kế một cửa sổ nhỏ, trên đó cứ mỗi giây, nếu có xuất hiện câu hỏi trắc nghiệm ngắn cùng với câu trả lời thì chúng sẽ hiển thị trong khoảng thời gian 1/80 cuối cùng của giây. Những học viên thực hiện chương trình học nghiêm túc sẽ tiếp nhận được các câu hỏi thi trong tình trạng vô thức và sẽ dễ dàng vượt qua kỳ thi hoặc kiểm tra. Ngân hàng đề có n câu hỏi trắc nghiệm và câu thứ i phải được hiển thị Ci lần. Để củng cố kiến thức và tránh sự đơn điệu làm não bộ nhàm chán, việc hiển thị câu hỏi phải đảm bảo trong m giây liên tiếp bất kỳ không có 2 câu hỏi nào giống nhau.
Yêu cầu: Hãy xác định khoảng thời gian tối thiểu (tính theo giây) thực hiện yêu cầu trên.

Dữ liệu: Vào từ file văn bản EL.INP:
• Dòng đầu tiên chứa 2 số nguyên cách nhau một dấu cách n và m (1<=m<=n<=100),
• Dòng thứ i trong n dòng sau chứa số nguyên Ci(0<Ci<=100)
Các số trên một dòng cách nhau một dấu cách.
Kết quả: Đưa ra file văn bản EL.OUT một số nguyên – khoảng thời gian tìm được.
Ví dụ:

EL.INP
3 2
2
3
1
EL.OUT
6

Với VD trên, ta có n=3 câu hỏi trắc nghiệm, và số lần phải hiển thị là
Câu hỏi 1(c1): 2 lần
Câu hỏi 2(c2): 3 lần
Câu hỏi 3:(c3): 1 lần
Yêu cầu của VD là sắp lịch sao cho trong m=2 giây liên tiếp bất kỳ ko có 2 câu hỏi nào giống nhau

Với VD trên ta có thể có các đáp án như sau
1 đáp án hợp lệ : c2 nghỉ c2 c1 nghỉ c2 c1 c3 : 8 giây
1 đáp án ko hợp lệ : c1 c2 c3 **c2 c2** c1
1 đáp án tối ưu : c1 c2 c1 c2 c3 c2: 6 giây

```
/* Code của @vietduc
ban đầu sort nó theo thứ tự giảm dần, có 100 số nên chắc không
cần qicksort
thứ tự như vậy chính là thứ tự ưu tiên xếp vào dãy kết quả, dùng
1 mảng độ rộng là m để kiểm tra sự xuất hiện của câu hỏi thứ i
đã xếp vào đó, ta tiến hành đùn data vào vào trước ra trước cứ
làm thế cho đến lúc số các câu là 0
*/

#include <iostream>
#include <fstream>
using namespace std;
int n,m;
struct nodevong
{
    int ci;
    nodevong* next;
    nodevong(){ci=-1;next=NULL;}
};
struct nodemang
{
    nodemang(){ed=true;}
    bool ed;
    int count;
};
void tao_vong(nodevong*&l,int m)
{
    nodevong* t,*v;
    for (int i=1;i<=m;i++)
    {
        t=new nodevong;
        if (l==NULL)
            v=l=t;
        else
            l=l->next=t;
    }
    l->next=v;
}
int tinhthoigian(nodevong*&l,nodemang*&c)
{
    int OLP=0,nn=n;
    while (nn)
    {
        if(l->ci >= 0  && c[l->ci].count>0 )
            c[l->ci].ed=true;
        int k=0;
        while (!c[k].ed) k++;
        //cout << k<<endl;
```

```
            l->ci=k;
            l=l->next;
            if (k<n)
            {
                c[k].count--;
                if (!c[k].count)
                    nn--;
                c[k].ed=false;
            }
            OLP++;
        }
        return OLP;
}

int main()
{
        ifstream infile("el.txt");
        infile >> n >>m;
        nodemang *c;
        c=new nodemang[n+1];
        for(int i=0; i<n; i++)
            infile>>c[i].count;
        // sap xep
        for (i=0; i<n-1; i++)
            for (int j=i+1; j<n;j++)
                if (c[i].count<c[j].count)
                {
                    nodemang t=c[i];
                    c[i]=c[j];
                    c[j]=t;
                }
        nodevong* l=NULL;
        tao_vong(l,m);
        cout << tinhthoigian(l,c)<<endl;
        return 0;
}
```

Bài: Dãy số (Ko chuyên 2009)

Cho dãy số gồm n số nguyên a1, a2, …, an. Tìm giá trị lớn nhất của hàm
f (i,j,k)= ai +2× aj +3× ak với $1 \le i < j < k \le n$.
Ví dụ: với dãy gồm 5 số -1, 2, -2, -3, 5 thì f (1,2,5)=-1+2×2+3×5=18 là lớn nhất.
Dữ liệu: Vào từ file văn bản SEQUENCE.INP:
• Dòng đầu tiên chứa số nguyên n ($3 \le n \le 10^5$),
• Dòng thứ i trong n dòng tiếp theo chứa số nguyên ai ($|ai| \le 10^9$).
Kết quả: Đưa ra file văn bản SEQUENCE.OUT một số nguyên – giá trị lớn nhất của hàm f (i,j,k)tìm được.
Ví dụ:
SEQUENCE.INP
5

-1
2
-2
-3
5
SEQUENCE.OUT
18

```
// Code của @Sounj
// Dùng : Quy Hoạch Động
#include <iostream>
#include <conio.h>
#define MAX 100000
using namespace std;
int max_f(int a[], int N)
{
    if(N<3) return 0;

    int f[MAX];
    int luu, tg;
    int i;

    // vòng lặp đầu tiên: f = ai
    f[0] = a[0];
    for(i=1; i<N; i++)
        if(f[i-1]<a[i]) f[i]=a[i]; else f[i] = f[i-1];

    // vòng lặp thứ 2: f = ai + 2* a[j]
    luu = f[1]; f[1] = f[0] + a[1]*2;
    for(i=2; i<N; i++)
    {
        tg = f[i];
        if(f[i-1]<luu+a[i]*2) f[i] = luu+a[i]*2; else f[i] = f[i-1];
        luu = tg;
    }

    // vòng lặp thứ 3: f = ai + 2* a[j] + 3*a[k]
    luu = f[2]; f[2] = f[1] + a[2]*3;
    for(i=3; i<N; i++)
    {
        tg = f[i];
        if(f[i-1]<luu+a[i]*3) f[i] = luu+a[i]*3; else f[i] = f[i-1];
        luu = tg;
    }

    return f[N-1];
}

void main()
{
    int a[] = {-1, 2, -2, -3, 5};
    int N = 5;

    cout<<"Ket qua: "<< max_f(a, N);
```

```
    getch();
}


// Code của @hunterphu
// Dùng : Quy Hoạch Động
#include<iostream>
#define k -1000000000
using namespace std;
long long a[100001];
int n;
long long x=k,y=k,z=k;
int main()
{
    scanf ("%d",&n);
    for (int i=1;i<=n;i++)
    {
        scanf ("%I64d",&a[i]);
        x=max(x,y+3*a[i]);
        y=max(y,z+2*a[i]);
        z=max(z,a[i]);
    }
    printf ("%I64d",x);
    system("pause");
    return 0;
}
```

Bài :Kết bạn (Ko chuyên 2009)

Theo quan niệm của người Á Đông cổ, mỗi cá nhân khi sinh ra đều ứng với một ngôi sao, được gọi là sao chiếu mệnh. Các hoạt động của cá nhân đều bị chi phối bởi ngôi sao này, kể cả quá trình kết bạn – hẹn hò. Theo thuyết Âm dương – Ngũ hành, hai người chỉ có thể tạo lập mối quan hệ bền vững khi các sao chiếu mệnh của họ không có các thuộc tính tương khắc. Qua hàng nghìn năm quan sát và chiêm nghiệm, các chiêm tinh gia đã ghi nhận được n sao và hầu hết các tính chất tương sinh – tương khắc giữa chúng. Để có thể nhanh chóng đáp ứng nhu cầu kiểm tra độ tương hợp của các sao, hiệp hội ABS (Association of Broker for Single) tạo lập cơ sở dữ liệu ghi nhận tính chất của tất cả các sao đã khảo sát được. Trong cơ sở dữ liệu này, các sao được đánh số từ 1 tới n; sao thứ i có một giá trị si thể hiện khả năng thích nghi của sao gọi là độ thích nghi. Hai sao khác nhau có thể có cùng độ thích nghi. Thông qua độ thích nghi của các sao, người ta xác định khả năng tương hợp của chúng. Khả năng tương hợp của 2 sao được tính bằng tổng 2 độ thích nghi của chúng.

Bài toán: Cho số nguyên dương n, dãy **s1, s2, …, sn** là độ thích nghi của các sao và số nguyên B. Hãy xác định số lượng các cặp sao (i, j) mà si + sj = B, với $1 \le i < j \le n$.

Ví dụ: trong 5 sao với độ thích nghi **3, 5, 6, 5, 3** có 4 cặp có khả năng tương hợp bằng 8.

Dữ liệu: Vào từ file văn bản FRIEND.INP:

• Dòng đầu tiên ghi 2 số nguyên n, B **($2 \le n \le 10^5$, $|B| \le 10^9$),**

• Mỗi dòng trong n dòng tiếp theo ghi một số nguyên là độ thích nghi của một sao, độ thích nghi có trị tuyệt đối bé hơn 2^15

.
Hai số trên cùng dòng cách nhau ít nhất một dấu cách.
Kết quả: Đưa ra file văn bản FRIEND.OUT một số nguyên – số lượng cặp sao có độ tương hợp B tìm được.

Ví dụ

FRIEND.INP
5 8
3
5
6
5
3
FRIEND.OUT
4

```
// Code của @hienclubvn
#include<stdio.h>
#include<stdlib.h>
#define IN "FRIEND.INP"
#define OUT "FRIEND.OUT"
long *A;
int n,B,count=0;
void Read()
{
      FILE *fin;
      fin=fopen(IN,"r");
      fscanf(fin,"%d %d",&n,&B);
      // Cap Phat bo nho dong
      A=(long*)malloc(n*sizeof(long));
      int i;
      for(i=0;i<n;i++)
         fscanf(fin,"%d",&A[i]);
      fclose(fin);
}
void Write()
{
      FILE *fout;
      fout=fopen(OUT,"w");
      fprintf(fout,"%d\n",count);
      fclose(fout);
}
void Process()
{
      int i,j;
      for(i=0;i<n-1;i++)
         for(j=i+1;j<n;j++)
           if(A[i]+A[j]==B) count++;
}
int main()
{
    Read();
    Process();
```

```
    Write();
    return 0;
}
```

```
// Code của @Vibzz90
#include<iostream>
#include<fstream>
using namespace std;
const char in[]="bai1in.txt";
const char out[]="bai1out.txt";
int *a;
long n,B;
int sosanh(void const *a1, void const *a2)
{
    return (*(int*)a1-*(int*)a2);
}
int process()
{
    int dem=0;
    long buf;
    qsort(a,n,sizeof(int),sosanh);
    for(int i=0;i<n;i++){
        for(int j=i+1;j<n;j++)
        {
            buf=a[i]+a[j];
            if(buf==B) dem++;
            if(buf>B) break;
        }
    }
    return dem;
}
void input()
{
    ifstream f(in);
    f>>n>>B;
    a=new int[n];
    for(int i=0;i<n;i++) f>>a[i];
    f.close();
}
void gentest()
{
    cout<<"Nhap n,B: ";
    cin>>n>>B;
    ofstream f(in);
    f<<n<<" "<<B<<endl;
    srand(time(NULL));
    for(int i=0;i<n;i++)
        f<<(rand()+rand()-32768)<<endl;
    f.close();
}
int main()
{
    cout<<"gentest?";
```

```
    int choice;cin>>choice;
    if(choice) gentest();
    input();
    cout<<process()<<endl;
    system("pause");
    return 0;
}
```

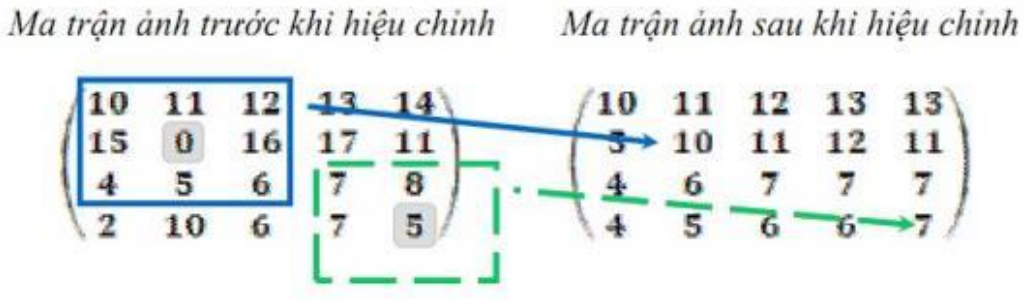
Bài: Hiệu chỉnh ảnh đơn sắc (Ko chuyên 2009)

Ảnh đơn sắc là ảnh chỉ gồm một màu nhưng các vùng trên ảnh khác nhau về mức sáng – ví dụ ảnh xám (grayscale image). Để biểu diễn một ảnh đơn sắc hình chữ nhật trên máy tính, người ta thường dùng một ma trận P, giá trị tại dòng i cột j của P chính là mức sáng của điểm ảnh tại vị trí tương ứng trên ảnh.

Việc chụp và đưa ảnh vào máy tính thỉnh thoảng có sai sót tạo nên nhiễu. Nhiễu là các điểm ảnh có độ sáng khác hẳn vùng ảnh xung quanh. Có nhiều cách làm giảm sự khác biệt này. Một trong những cách đó là dùng một cửa sổ hình vuông 3x3 có cạnh song song với cạnh của ảnh và hiệu chỉnh các điểm ảnh trong vùng ảnh bị nhiễu. Mỗi điểm ảnh ở dòng i cột j sẽ được thay thế bằng trung vị của các giá trị ảnh đang có trong cửa sổ có tâm tại vị trí (i, j) ở ảnh gốc ban đầu. Trong các trường hợp điểm ảnh ở biên, chỉ xét trung vị của các giá trị nằm trong ảnh.

Nhắc lại rằng, trung vị của k số a1, a2, … ak là số ở vị trí t khi sắp xếp k số này theo trật tự tăng dần, trong đó t là phần nguyên của số (k+1)/2

Dưới đây là ví dụ mô tả việc hiệu chỉnh ảnh bằng cách nêu trên.



Bài toán: Cho ma trận số nguyên P cấp m× n biểu diễn một vùng ảnh đơn sắc có nhiễu. Hãy dùng cách đã nêu ở trên để hiệu chỉnh các điểm ảnh trong vùng ảnh bị nhiễu.

Dữ liệu: Vào từ file văn bản ADJUST.INP gồm:

• Dòng đầu tiên chứa 2 số nguyên m, n (1 ≤m, n ≤ 100),

• m dòng tiếp theo mỗi dòng ghi n số nguyên không âm là mức sáng các điểm ảnh. Giá trị mức sáng không vượt quá 255.

Kết quả: Đưa ra file văn bản ADJUST.OUT gồm m dòng, mỗi dòng gồm n số là các mức sáng trong vùng ảnh sau khi đã hiệu chỉnh.

Hai số trên cùng dòng cách nhau ít nhất một dấu cách.

Ví dụ:

ADJUST.INP

4 5

10 11 12 13 14

15 0 16 17 11
4 5 6 7 8
2 10 6 7 5

ADJUST.OUT
10 11 12 13 13
5 10 11 12 11
4 6 7 7 7
4 5 6 6 7

```
// Code của @hienclubvn
#include<stdio.h>
#include<stdlib.h>
#define IN "ADJUST.INP"
#define OUT "ADJUST.OUT"
int **A,**T,m,n;
void Input()
{
      FILE *fin;
      fin=fopen(IN,"r");
      int i,j;
      fscanf(fin,"%d %d",&m,&n);
      // Cap phat dong
      A=(int**)malloc(m*sizeof(int));
      for(i=0;i<m;i++) A[i]=(int*)malloc(n*sizeof(int));
      T=(int**)malloc(m*sizeof(int));
      for(i=0;i<m;i++) T[i]=(int*)malloc(n*sizeof(int));
      // read File
      for(i=0;i<m;i++)
        for(j=0;j<n;j++) fscanf(fin,"%d",&A[i][j]);
      fclose(fin);
}
void Sort(int *B,int n)
{
      int i,j;
      for(i=0;i<n-1;i++)
        for(j=i+1;j<n;j++)
            if(B[i]>B[j])
            {
               int temp=B[i];
               B[i]=B[j];
               B[j]=temp;
            }
}
int FindPoint(int i,int j)
{
   int B[10];
   int k,l,m1,n1,t=0;
   int di[3]={-1,0,1};
   int dj[3]={-1,0,1};
   for(k=0;k<3;k++)
   {
     m1=i+di[k];
```

```
      for(l=0;l<3;l++)
      {
         n1=j+dj[l];
         if(m1>=0&&m1<m&&n1>=0&&n1<n)
             B[t++]=A[m1][n1];
      }
    }
    Sort(B,t);
    int p=(t+1)/2-1;
    int x=B[p];
    return x;
}
void Process()
{
      int i,j;
      for(i=0;i<m;i++)
      for(j=0;j<n;j++)
      {
          T[i][j]=FindPoint(i,j);
      }
}
void Output()
{
     FILE *fout;
     fout=fopen(OUT,"w");
     int i,j;
     for(i=0;i<m;i++)
     {
        for(j=0;j<n;j++)
        if (j!=n-1)fprintf(fout,"%d ",T[i][j]);
        else  fprintf(fout,"%d\n",T[i][j]);
     }
     fclose(fout);
}
int main()
{
     Input();
     Process();
     Output();
     free(A);
     free(T);
     return 0;
}
```