**CHUYÊN ĐỀ 1.**

**BÀI TOÁN TỔNG ƯỚC, ĐẾM ƯỚC CỦA SỐ N**

**1. Áp dụng công thức**

**n=p1a1p2a2…pkak**

***a. Số lượng ước của n***

**d=(a1+1)(a2+1)…(ak+1);**

***b. Tổng các ước của n***

**S=**

*int\_fast64\_t tonguoc(int\_fast64\_t n)*

*for(int i=2;i<=sqrt(n);i++)*

*if(n%i==0)*

*{*

*p[++k]=i;*

*a[k]=0;*

*while(n%i==0)*

*{*

*a[k]++;*

*n/=i;*

*}*

*}*

*if(n>1)*

*{*

*p[++k]=n;*

*a[k]=1;*

*}*

**d=(a1+1)(a2+1)…(ak+1); // Số lượng ước của n**

*int\_fast64\_t demuoc(int\_fast64\_t n)*

*{*

*int\_fast64\_t k=0,ans=1, p[100], a[100];*

*for(int i=2;i<=sqrt(n);i++)*

*if(n%i==0)*

*{*

*p[++k]=i;*

*a[k]=0;*

*while(n%i==0)*

*{*

*a[k]++;*

*n/=i;*

*}*

*}*

*if(n>1)*

*{*

*p[++k]=n;*

*a[k]=1;*

*}*

*for(int i=1;i<=k;i++) ans\*=(a[i]+1); return ans;*

*}*

**S= // Tổng các ước của n**

*int\_fast64\_t tonguoc(int\_fast64\_t n)*

*{*

*int\_fast64\_t k=0,ans=1, p[100], a[100];*

*for(int i=2;i<=sqrt(n);i++)*

*if(n%i==0)*

*{*

*p[++k]=i;*

*a[k]=0;*

*while(n%i==0)*

*{*

*a[k]++;*

*n/=i;*

*}*

*}*

*if(n>1)*

*{*

*p[++k]=n;*

*a[k]=1;*

*}*

*for(int i=1;i<=k;i++)*

*ans\*=(((powl(p[i],a[i]+1)))-1)/(p[i]-1); return ans;*

*}*

**2. Dùng mảng lưu trữ**

***a. Tính tổng ước của n <1000000 dùng mảng tổng***

*for ( i = 1; i <= 1000000; ++ i) {*

*for ( j = i; j <= 1000000; j += i) sum[j] += i;*

***b. Đếm các ước của n <1000000 dùng mảng đếm***

*for ( i = 1; i <= 1000000; ++ i) {*

*for ( j = i; j <= 1000000; j += i) d[j] ++;*

***Lưu ý: dùng mảng lưu trữ phù hợp với những bài toán lặp gọi thao tác nhiều lần***

***Ví dụ:***

**Bài 3. (3,0 điểm) Tổng ước chẵn**

Cho số nguyên dương 𝑛, ước nguyên dương của 𝑛 là số 𝑖 = 1, 2, … , 𝑛 thỏa mãn: 𝑛 chia

hết cho 𝑖.

**Yêu cầu:** Hãy lập trình đếm số các số nguyên dương trong đoạn [𝐿, 𝑅] có tổng các ước là một

số chẵn.

**Dữ liệu vào:**

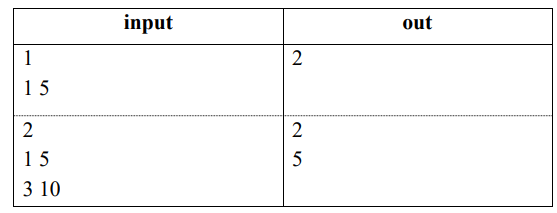
• Dòng đầu ghi số nguyên dương 𝑞 (1 ≤ 𝑞 ≤ 106);

• 𝑞 dòng tiếp theo mỗi dòng ghi hai số nguyên 𝐿, 𝑅 (1 ≤ 𝐿 ≤ 𝑅 ≤ 106)

**Kết quả:**

• Ghi ra 𝑞 dòng, mỗi dòng ghi ra số lượng số nguyên dương trong đoạn [𝐿, 𝑅] thỏa mãn có tổng các ước là một số chẵn.

**Ví dụ:**

****

• Ràng buộc 1: có 25% số test của bài ứng với 25% số điểm của bài có 𝑞 = 1, 𝑛 ≤ 1000;

• Ràng buộc 2: có 25% số test của bài ứng với 25% số điểm của bài có 𝑞, 𝑛 ≤ 1000;

• Ràng buộc 3: có 25% số test của bài ứng với 25% số điểm của bài có 𝑛 ≤ 5000;

• Ràng buộc 4: có 25% số test của bài ứng với 25% số điểm của bài có 𝑞, 𝑛 ≤ 106

Ý tưởng của thuật toán 4 giống với ý tưởng của thuật toán sáng nguyên tố (học sinh cần nắm vững ***thuật toán sàng nguyên tố***) và ***tổng cộng dồn.***

**Cụ thể như sau:**

- Gọi 𝑆[𝑗] là tổng các ước của số nguyên j,

- Thay vì với mỗi j ta tính tổng S[j] thì ta sẽ làm ngược lại,

- Với mỗi số nguyên i ta sẽ tích lũy i vào tổng S[j] với j là bội của i.

***for (long long i = 1; i <= 1000000; ++ i) {***

***for (long long j = i; j <= 1000000; j += i) {***

***sum[j] += i;***

***}***

***cnt[i] = cnt[i - 1];***

***if (S[i] % 2 == 0) {***

***cnt[i] += 1;***

***}***

}

**for(int t = 1; t<=q; t++){**

**int L, R;**

**cin>>L>>R;**

**cout<<cnt[R] – cnt[L-1]<<endl;**

}

*Độ phức tạp của thuật toán 𝑂(𝑞 + 𝑛 ∗ log2 𝑛); 𝑣ớ𝑖 𝑛 ≤ 106 => thuật toán thành công*

**Sàng nguyên tố: Liệt kê các số nguyên tố nhỏ hơn n**

#include<bits/stdc++.h>

using namespace std;

bool snt[100000001];

*int i,n;*

*int main()*

*{*

*cin>>n;*

*memset(snt,true,sizeof(snt));*

*snt[1]=false;*

*int p=2;*

*while(p\*p<n)*

*{*

**for(i=2;i<=n/p;i++) snt[i\*p]=false;**

*for(p++;p\*p<n&& !snt[p];p++);*

*}*

*for(i=1;i<=n;i++) if(snt[i]) cout<<i<<" ";*

*}*

*}*

**Tạo sàng nguyên tố:**

memset(ngto,true,100000);

ngto[0]=0;

ngto[1]=0;

for(i=2;i<=sqrt(32000);i++)

if(ngto[i])

for(int j=i\*i;j<=32000;j+=i) ngto[j]=false;

**LUYỆN ĐỀ**

**Bài 1. Tính tổng các ước**

Cho hai số nguyên dương a,b. ính tổng tất cả các số nguyên x thỏa mãn

* + x là ước của a
  + 3x là ước của b

Dữ liệu vào: file **tonguoc.inp** 2 số nguyên duong a và b (1a,b 1012)

Dữ liệu ra: file **tonguoc.out** là tổng tấy cả các số x thỏa mãn điều kiện trên kết quả không quá 1018 nếu không có x thỏa mãn kết quả 0

|  |  |  |
| --- | --- | --- |
| **tonguoc.inp** | **tonguoc.out** | **Giải thích** |
| 4 18 | 3 | x=1,x=2 |
| 1 2 | 0 | Không có x thỏa mãn |

**Bài 2. Song nguyên tố**

Song nguyên tố là số có tổng các ước là số nguyên tố

Yêu cầu: Hãy đếm số lượng các số song nguyên tố trong đoạn [a;b]

**Dữ liệu vào:songnt.inp** dòng đầu ghi số nguyên T là số lượng đoạn cần đếm, T dòng tiếp theo ghi các số a,b(ab) cách nhau 1 dấu cách

**Dữ liệu ra:songmt.out** gồm T dòng ghi kết quả của T đoạn

**Ví dụ**

|  |  |
| --- | --- |
| **Songnt.inp** | **Songnt.out** |
| **2**  **2 5**  **1 100** | **4**  **32** |

**BÀI 3: SỐ TỰ NHIÊN NHỎ NHẤT**

Nam một người bạn của Nguyên đang tìm cách giải một bài toán liên quan tới số tự nhiên và cần sự giúp đỡ của Nguyên, nhưng thử thách lần này là một dãy gồm N số tự nhiên bất kỳ nằm trong đoạn từ 0 tới 109, tìm số tự nhiên nhỏ nhất không có trong dãy số đó. Vì số lượng các số tự nhiên trong dãy số đã cho có thể lên tới 106 phần tử nên việc tìm thủ công là không thể mà cần một thuật toán để cài đặt vào máy tính và nhờ máy tính tìm giúp.

**Yêu cầu:** Cho một dãy A gồm N (1 ≤ N ≤ 106) số tự nhiên. Hãy tìm số tự nhiên nhỏ nhất không xuất hiện trong dãy A.

**Bài 4. Trò chơi**

Nhân dịp lễ giáng sinh, công viên trung tâm tổ chức trò chơi "con số may mắn". Mỗi em nhỏ đến tham dự sẽ được phát một số nguyên dương. Công viên có một thiết bị quay số, mỗi lần quay sẽ tạo ngẫu nhiên một số nguyên dương có giá trị tuyệt đối không vượt quá 104. Người dẫn chương trình sẽ thực hiện N lần quay số. Số nào xuất hiện nhiều nhất trong N lần quay được gọi là con số may mắn và em nhỏ nào có con số may mắn thì sẽ được phần thưởng.

**Yêu cầu:** Cho N con số xuất hiện trong N lần quay. Bạn hãy giúp người dẫn chương trình xác định số lần xuất hiện của con số may mắn.

Dữ liệu vào từ file văn bản Bai1.inp:

• Dòng đầu là số N (1 ≤ N ≤ 104).

• Dòng tiếp theo có N số là các số xuất hiện trong N lần quay.

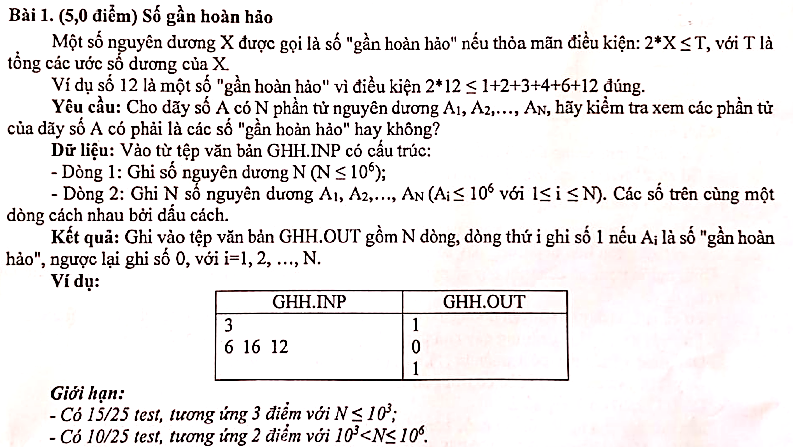
Kết quả ghi ra file văn bản Bai1.out: Là số lần xuất hiện của con số may mắn.

**BÀI 5: SỐ THÂN THIỆN**

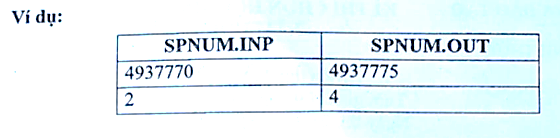
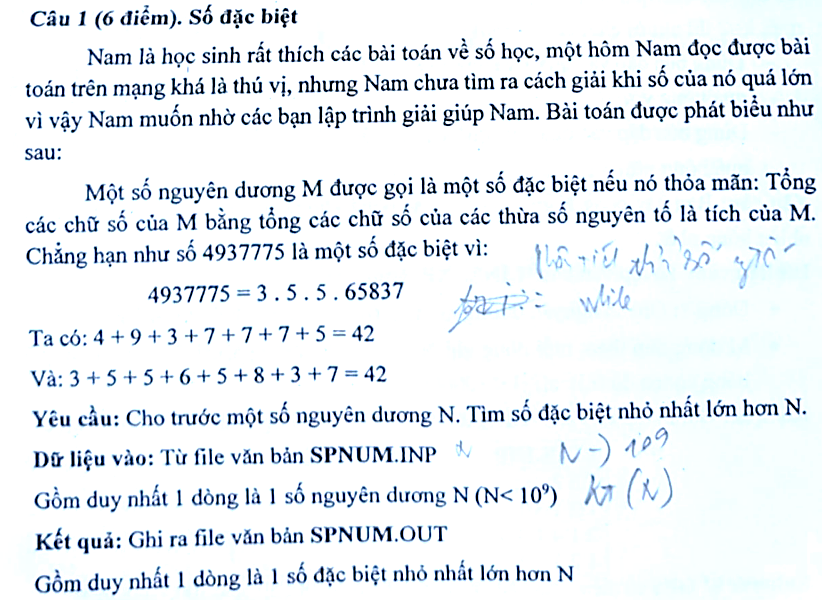
Đang tìm hiểu các thuật toán về số tự nhiên, Nguyên phát hiện ra số tự nhiên có rất nhiều tính chất thú vị. Ví dụ số hoàn hảo có tính chất: tổng các ước bằng 2 lần số đó, như số 6, số 24… Nhiều số tự nhiên khi tìm ước chung lớn nhất với số đảo ngược của nó bằng 1, những số như thế được gọi là số thân thiện. Chẳng hạn số 23, số đảo ngược của nó là 32, hai số này có ước chung lớn nhất là 1 nên số 23 là số thân thiện và 32 cũng là số thân thiện.

**Yêu cầu:** Cho 2 số tự nhiên a, b (10 ≤ a ≤ b ≤ 104). Hãy đếm xem trong đoạn từ a đến b có bao nhiêu số thân thiện.

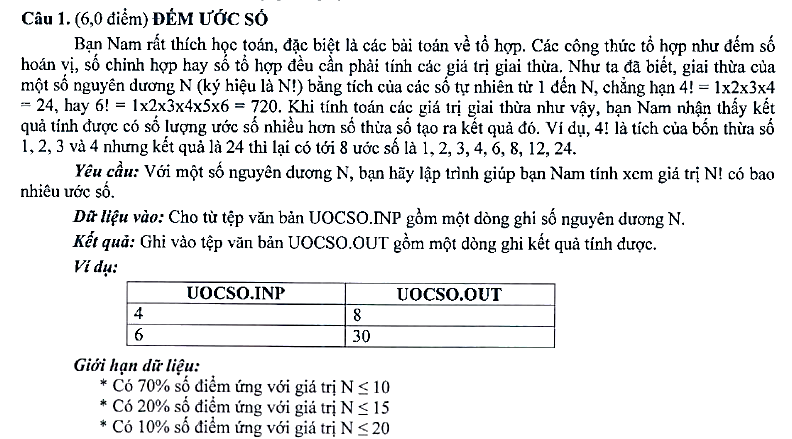
**Bài 6**

****

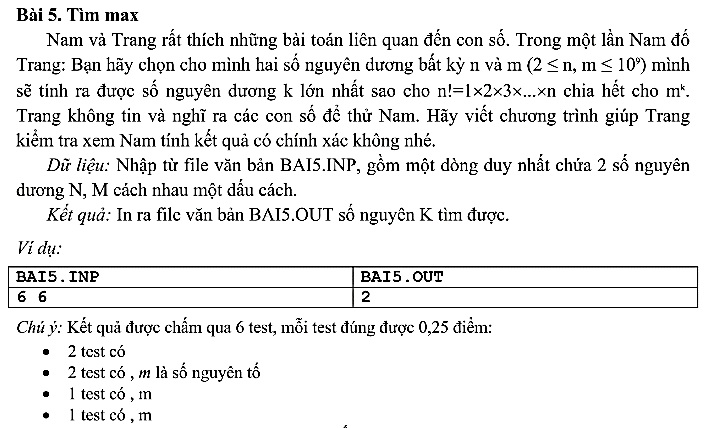
**Bài 7**

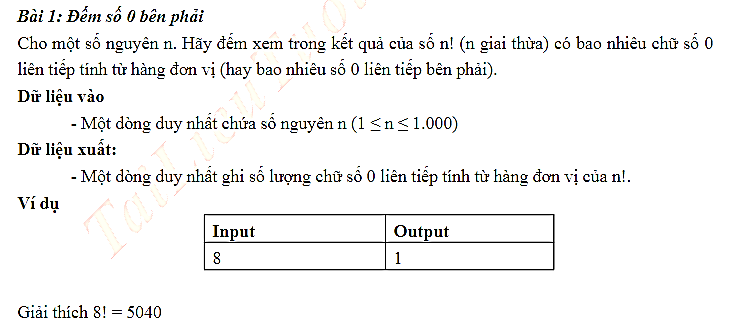
****

**Bài 8.**

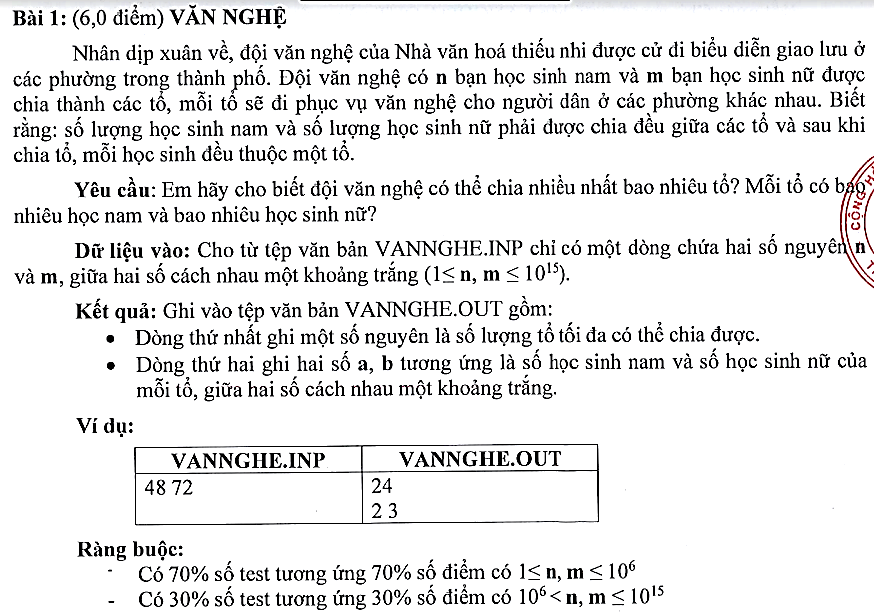
****

**Bài 9.**

****

**Bài 10.**

**Bài 11.**

****

**CHUYÊN ĐỀ 2. CHIA HẾT, ĐỒNG DƯ**

**1. DẤU HIỆU CHIA HẾT: S LÀ SÂU**

**Cho hết cho 2: s[s.size()-1]%2=0**

**Cho hết cho 3: (s0+s1+…+ s.size()-1)%3**

**Cho hết cho 4: hai chữ tận cùng chia hết cho 4**

**Cho hết cho 5: Chữ số tận cùng là 0 hoặc 5**

**Cho hết cho 6 chia hết cho 2 và 3**

**Chia hết cho 7: (..(((s0\*3)+s1)\*3+s2)\*3+…+s[s.size()-1]\*3)))..)+ s.size()-1)%7=0**

**Chia hết 8: 3 Chữ số cuối cùng chia hết cho 8**

**Chia hết cho 9: (s0+s1+…+ s.size()-1)%3**

**2. TÌM SỐ DƯ**

*a. Số nhỏ hơn 10 chữ số*

*Tìm số dư của a cho b*

**Có n=a/b**

**R là số dư cần tìm: R=a-n.b**

*b. Số lớn hơn 10 chữ số*

a=a1a2a3a4a5a6a7a8a9a10a11a12..an

Tìm phần dư của a1a2a3a4a5a6a7a8a9a10 với b được r1

a= r1 a11a12 a13a14 a15a16 a17a18 a19a20 a21a22..an

Tìm phần dư của r1 a11a12 a13a14 a15a16 a17a18 a19 với b được r2

a= r2 a20a21..an làm tương tự đến khi dc rn

**3. TÌM SỐ DƯ CỦA TỔNG, TÍCH LỚN**

**(a+b)%m=(a%m+b%m)%m**

**(a-b)%m=(a%m-b%m)%m**

**(a.b)%m=(a%m.b%m)%m**

**Đề 2022: Tìm số dư S=a+2a+..+na cho 109+7**

**S=a(1+2+..+n)**

#include<bits/stdc++.h>

using namespace std;

int main()

{

int\_fast64\_t a,n,ans,m=1000000007;

cin>>a>>n;

if(n%2==0)

{ans=((a%m)\*((n/2)%m)\*((n+1)%m))%m;

cout<<ans;

}

else

{

ans=((a%m)\*(n%m)\*(((n+1)/2)%m))%m;

cout<<ans;

}

}

**4. TÌM SỐ DƯ an CHO b**

**(an )%b = ?**

**k=n%(b-1)**

**x=a%b**

**(an )%b = (xk)%b**

**5. TÌM CHỮ SỐ TẬN CÙNG CỦA an**

* **Nếu a có tận cùng là 0;1;5;6 thì an cũng có tận cùng là: 0;1;5;6**
* **Nếu a tận cùng bằng 9: n chẵn: tận cùng bằng 1, n lẻ tận cùng bằng 9**
* **Nếu a tận cùng bằng 4: n chẵn: tận cùng bằng 6, n lẻ tận cùng bằng 4**
* **Nếu a có tận cùng là 2;3;7,8:**
* **Lấy n chia cho 4: n=4k+r với r={0;1;2;3}**

*#include <bits/stdc++.h>*

*using namespace std;*

*int\_fast64\_t tclt(string s,int\_fast64\_t k)*

*{*

*int\_fast64\_t n;*

*n=s[s.size()-1]-'0';*

*if( n==0 || n==5 || n==6 || n==1 && k!=0) return n; //0,1,5,6 dung voi moi TH*

*if( n==4 || n==9) //4,9 xet so mu chan hoac le*

*{*

*if(k%2==1) return n;*

*if(k%2==0)*

*{*

*if(n==4) return 6;*

*return 1;*

*}*

*}*

*//con lai 2,3,7,8*

*if( k%4==1 ) return n;*

*if( k%4==2 )*

*{*

*if( n==2 || n==8 ) return 4;*

*if( n==3 || n==7 ) return 9;*

*}*

*if( k%4==3 )*

*{*

*if( n==3 ) return 7;*

*if( n==7 ) return 3;*

*if( n==2 ) return 8;*

*if( n==8 ) return 2;*

*}*

*if( k%4==0 )*

*{*

*if( n==3 || n==7 ) return 1;*

*if( n==2 || n==8 ) return 6;*

*}*

*}*

*int main()*

*{*

*string s;*

*cin>>s;*

*int\_fast64\_t k;*

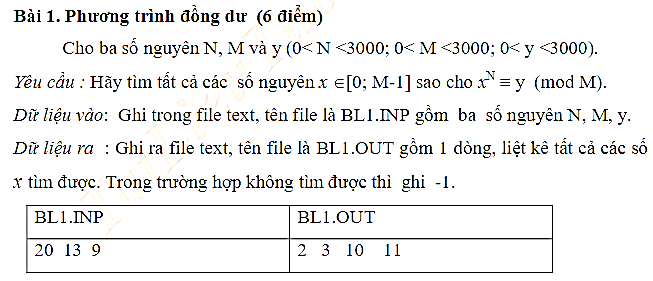
*cin>>k;*

*cout<<tclt(s,k)<<endl;*

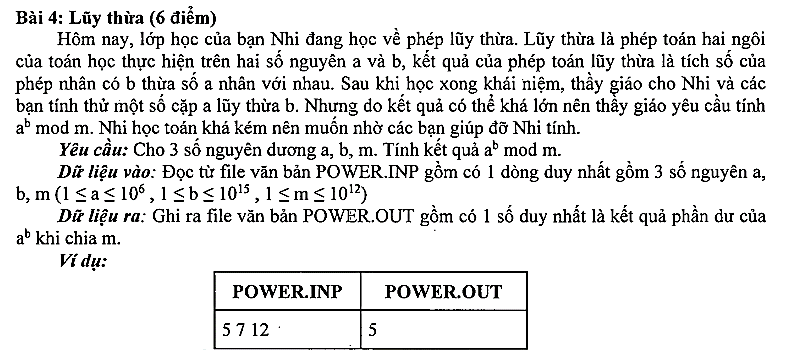
*}*

**LUYỆN ĐỀ**

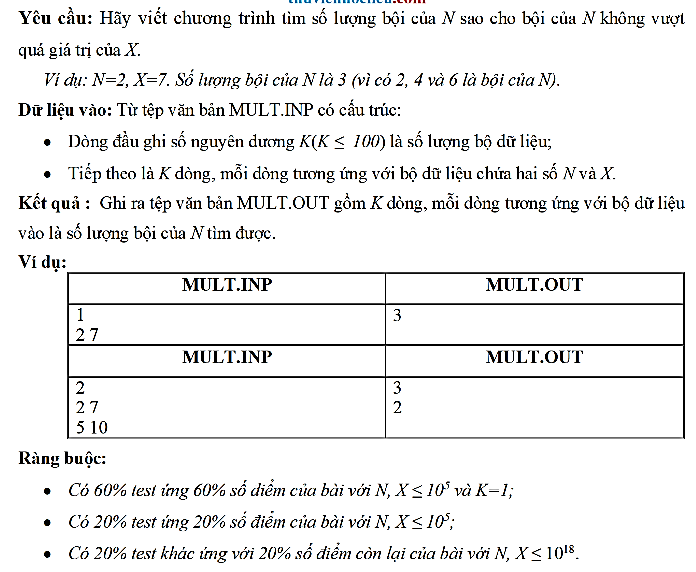
**Câu 1.**

****

**Câu 2.**

****

**Câu 3**



**Câu 4.Số chia hết**

Viết các số nguyên dương có tổng các chữ số chia hết cho số chữ số (trong biểu diễn thập phân) theo thư tự tăng ta thu được một dãy vô hạn :

1 2 3 4 5 6 7 8 9 11 13 15 17 19 20….

Cho số nguyên dương k, hãy chỉ ra chữ số thứ k của dãy số( chỉ số từ 1)

Dữ liệu: Số k(1 k 105

Kết quả:Số nguyên kế quả

|  |  |
| --- | --- |
| DIVN.INP | DIVN.OUT |
| 1 | 1 |
| 15 | 20 |

**CHUYÊN ĐỀ 3. DÃY CON LIÊN TIẾP CÓ TỔNG(=<> S)**

**//dãy con liên tiếp có tổng bằng S**

*#include <bits/stdc++.h>*

*using namespace std;*

*long long p=1;*

*int n,i,S, a[100];int sum[100];*

*int main()*

*{*

*sum[0]=0;*

*cin>>n>>S;*

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

*{*

*cin>>a[i];*

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

*}*

*int res=0;*

*sort(sum,sum+n+1);*

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

*{*

*if(binary\_seach(sum,i,n,sum[i]+S)) res++;*

*}*

*cout<<res;*

*}*

**2. Số dãy con liên tiếp có tổng lớn hơn S**

*#include <bits/stdc++.h>*

*using namespace std;*

*long long p=1;*

*int n,i,S, a[100];int\_fast64\_t sum[100];*

*int main()*

*{*

*sum[0]=0;*

*cin>>n>>S;*

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

*{*

*cin>>a[i];*

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

*}*

*int res=0;*

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

*{*

*int t=lower\_bound(sum,sum+n+1,sum[i]+S)-sum; cout<<"i="<<i<<" t="<<t<<endl;*

*if(t<n+1) res+=n-t+1;*

*}*

*cout<<res;*

*}*

**3.Số dãy con liên tiếp có tổng lớn hơn S và nhỏ hơn s**

*#include <bits/stdc++.h>*

*using namespace std;*

*long long p=1;*

*int n,i,S,s, a[100];int\_fast64\_t sum[100];*

*bool comp(int a, int b)*

*{*

*return a <= b;*

*}*

*int main()*

*{*

*sum[0]=0;*

*cin>>n>>s>>S;*

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

*{*

*cin>>a[i];*

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

*}*

*int res=0;*

*sort(sum,sum+n+1);*

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

*{*

*int t1=upper\_bound(sum,sum+n+1,sum[i]+s,comp)-sum;*

*int t2=lower\_bound(sum,sum+n+1,sum[i]+S)-sum;*

*if(sum[t2]!=S)t2=t2-1;*

*cout<<"i="<<i<<" t1="<<t1<<" t2="<<t2<<endl;*

*if(t2>t1) res+=t2-t1+1;*

*if(t1==t2)res++;*

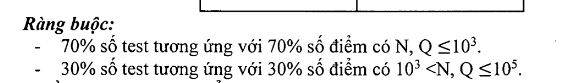
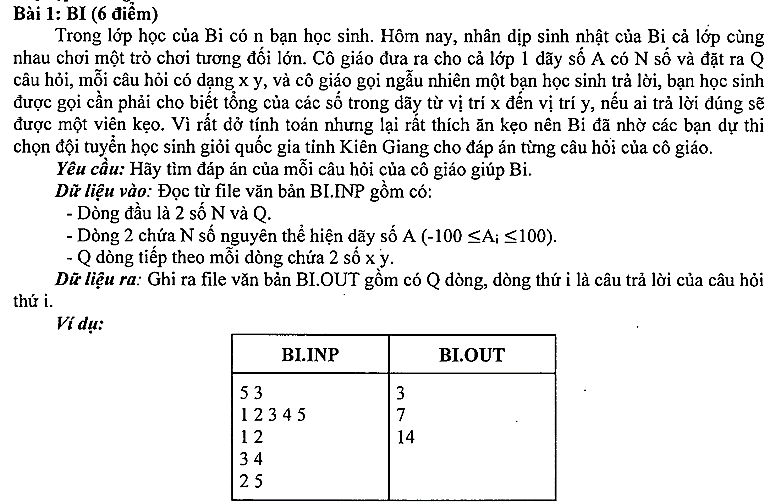
*cout<<" res"<<res<<endl;*

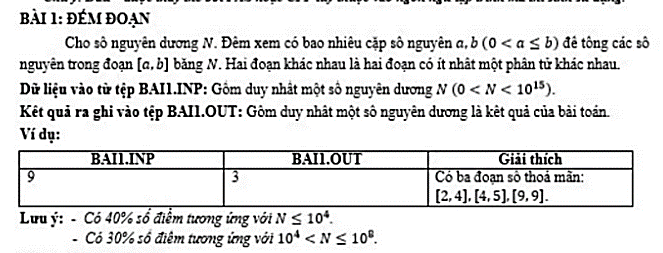
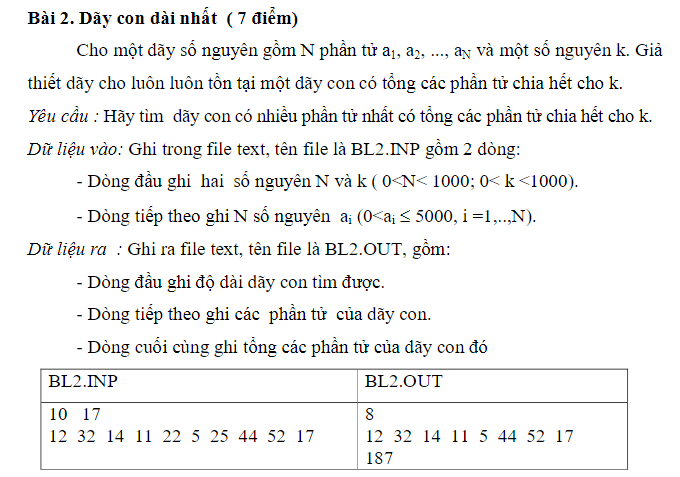
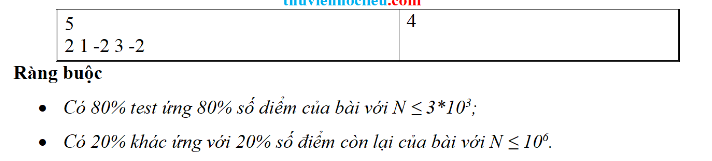
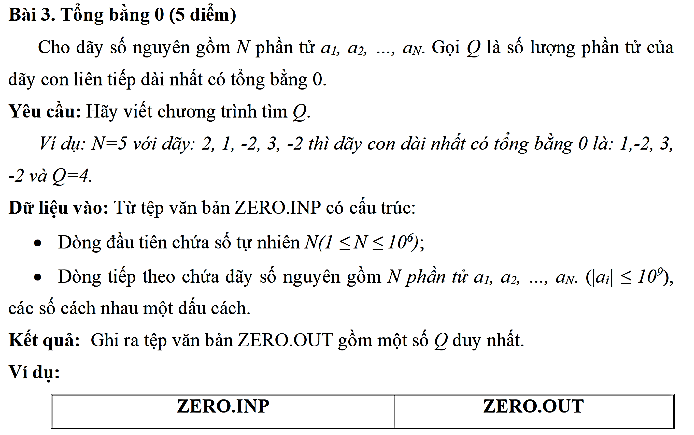
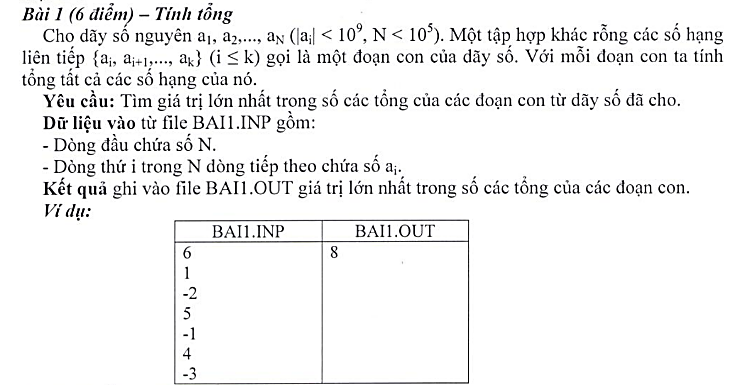
*}*

*cout<<res;*

*}*

**LUYỆN ĐỀ**

****

****

**Bài 2**

**Bài 5**

**Bài 4**

**CHUYÊN ĐỀ 4. MẢNG 2 CHIỀU**

**a. Nhập mảng**

*Mảng a có n hàng, m cột*

cin>>m>>n;

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

for(j=1;j<=m;j++) cin>>a[i][j];

**b. Xuất mảng**

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

{

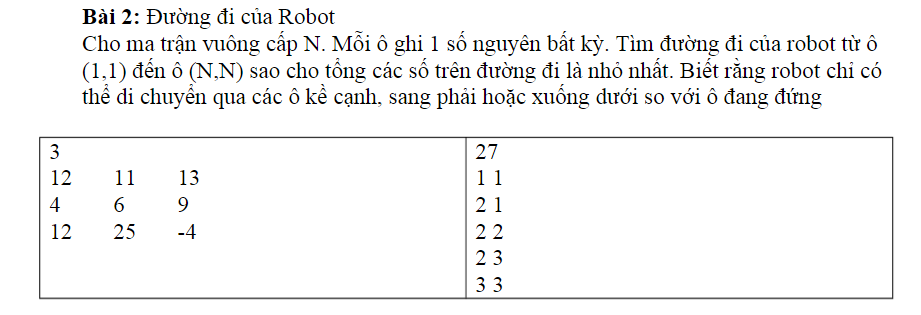
for(j=1;j<=m;j++) cout<<a[i][j]<<" ";

cout<<endl<<endl;

}

**c. Thao tác trên mảng:**

***Tìm đường đi thỏa mãn điều kiện:***



#include<bits/stdc++.h>

using namespace std;

long long i,j,m,n,a[1000][1000],b[1000][1000],ii[1000],jj[1000],k=0;

int main()

{

cin>>m>>n;

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

for(j=1;j<=m;j++) cin>>a[i][j];

b[1][1]=a[1][1];

for(i=2;i<=m;i++) b[i][1]=b[i-1][1]+a[i][1];

for(i=2;i<=n;i++) b[1][i]=b[1][i-1]+a[1][i];

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

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

b[i][j]=a[i][j]+ min(b[i-1][j],b[i][j-1]);

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

{

for(j=1;j<=m;j++) cout<<b[i][j]<<" ";

cout<<endl<<endl;

}

i=m;j=n; ii[1]=i;

jj[1]=j;

while(i>1||j>1)

{

k++;

ii[k]=i;

jj[k]=j;

if(b[i][j]==a[i][j]+b[i-1][j])i--; else j--;

}

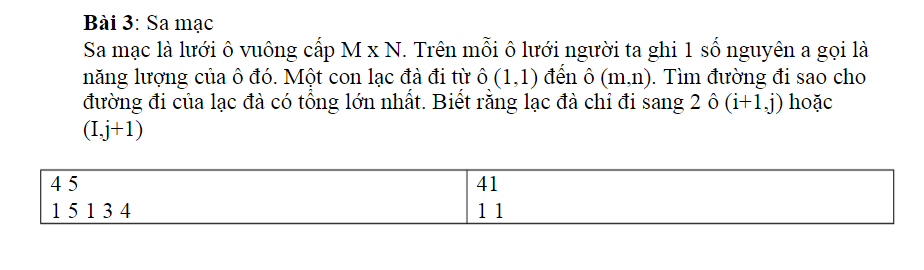
cout<<b[m][n]<<endl;

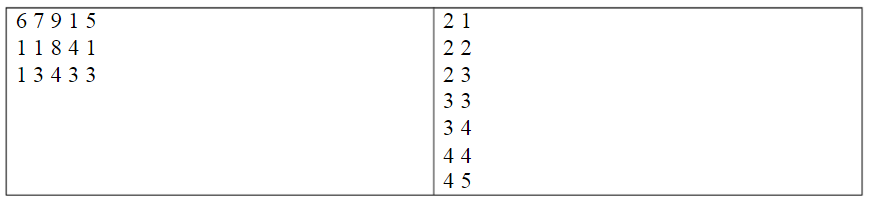
cout<<1<<" "<<1<<endl;

for(i=k;i>=1;i--) cout<<ii[i]<<" "<<jj[i]<<endl;

}

}





*#include<bits/stdc++.h>*

*using namespace std;*

*long long i,j,m,n,a[1000][1000],b[1000][1000],ii[1000],jj[1000],k=0;*

*int main()*

*{*

*cin>>m>>n;*

*for(i=1;i<=m;i++)*

*for(j=1;j<=n;j++) cin>>a[i][j];*

*b[1][1]=a[1][1];*

*for(i=2;i<=m;i++) b[i][1]=b[i-1][1]+a[i][1];*

*for(i=2;i<=n;i++) b[1][i]=b[1][i-1]+a[1][i];*

*for(j=2;j<=n;j++)*

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

*b[i][j]=a[i][j]+ max(b[i-1][j],b[i][j-1]);*

*for(i=1;i<=m;i++)*

*{*

*for(j=1;j<=n;j++) cout<<b[i][j]<<" ";*

*cout<<endl<<endl;*

*}*

*i=m;j=n; ii[1]=i;*

*jj[1]=j;*

*while(i>1||j>1)*

*{*

*k++;*

*ii[k]=i;*

*jj[k]=j;*

*if(b[i][j]==a[i][j]+b[i-1][j])i--; else j--;*

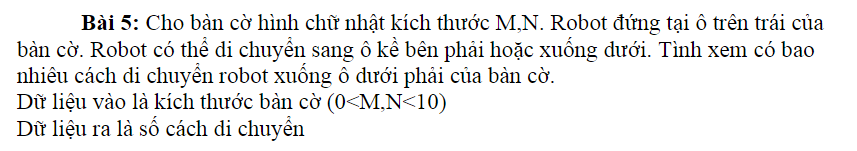
*}*

*cout<<b[m][n]<<endl;*

*cout<<1<<" "<<1<<endl;*

*for(i=k;i>=1;i--) cout<<ii[i]<<" "<<jj[i]<<endl;*

*}*



*#include<bits/stdc++.h>*

*using namespace std;*

*long long i,j,m,n,a[1000][1000],b[1000][1000],ii[1000],jj[1000],k=0;*

*int main()*

*{*

*cin>>m>>n;*

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

*for(j=1;j<=m;j++) cin>>a[i][j];*

*for(i=1;i<=m;i++) b[i][1]=1;*

*for(i=1;i<=n;i++) b[1][i]=1;*

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

*for(j=2;j<=n;j++)*

*b[i][j]=b[i-1][j]+b[i][j-1];*

*for(i=1;i<=m;i++)*

*{*

*for(j=1;j<=n;j++) cout<<b[i][j]<<" ";*

*cout<<endl<<endl;*

*}*

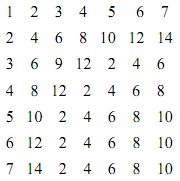
*}*

**LUYỆN ĐỀ:**

**Câu 1** (6 điểm):

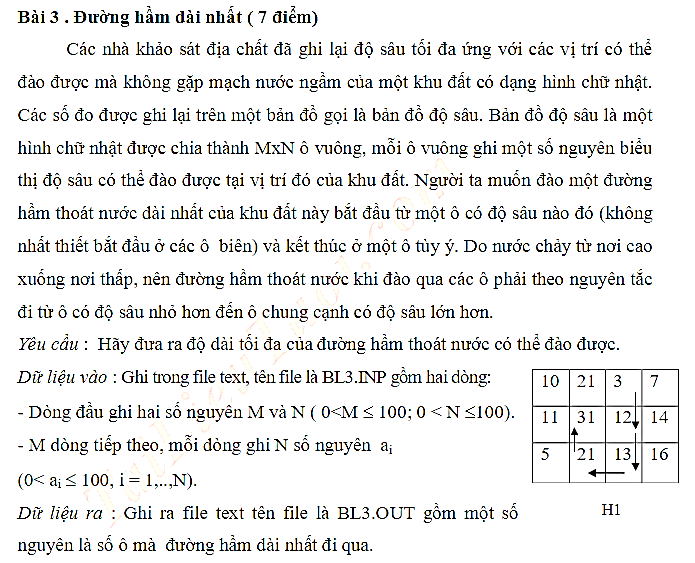
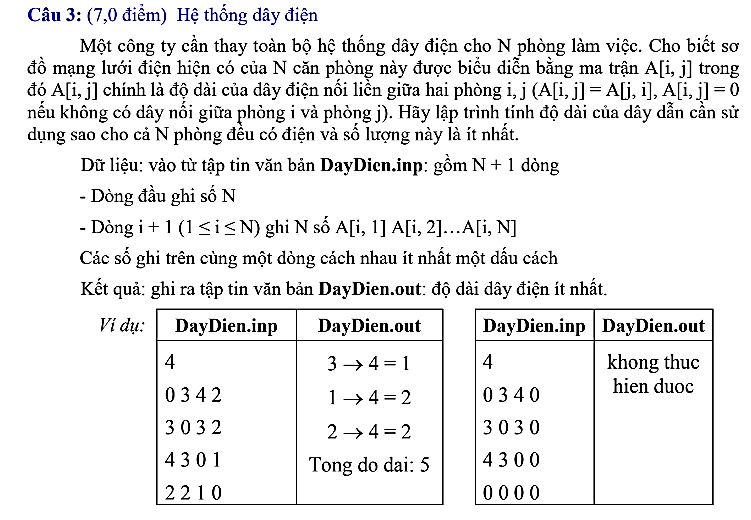
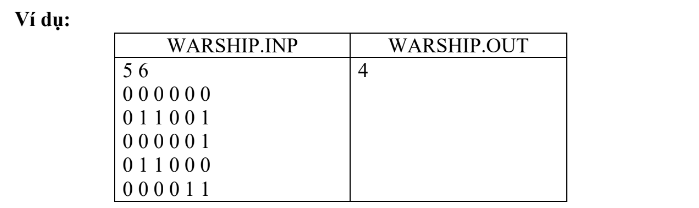
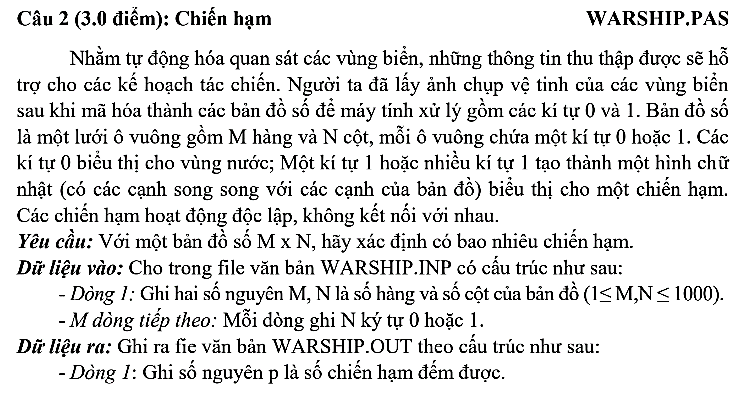
Cho một số nguyên dương N bất kì (N ≤ 20).

Hãy viết chương trình tạo mảng N x N phần tử nguyên dương (giá trị lớn nhất trong bảng là 2\*N) theo quy luật cho trong ví dụ sau với N = 7

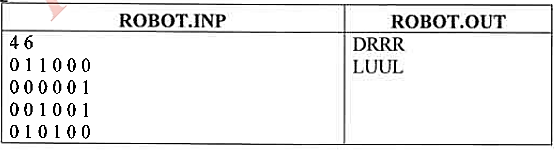
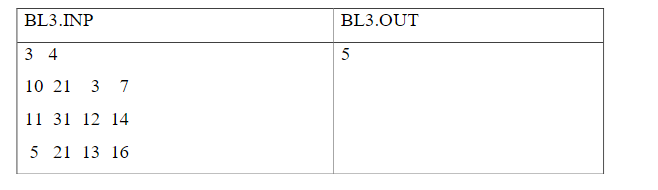
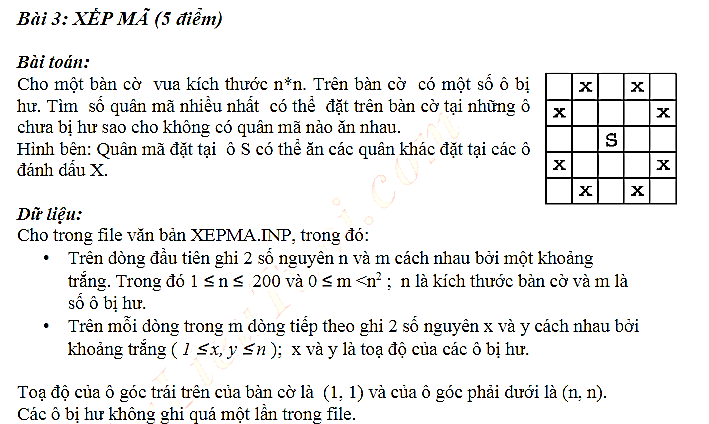
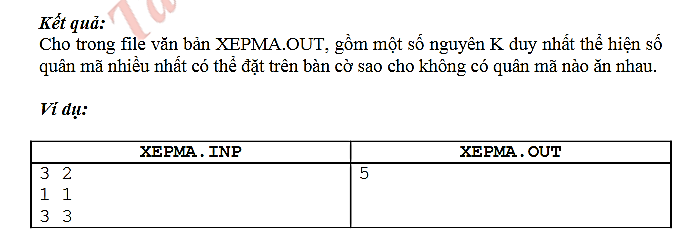
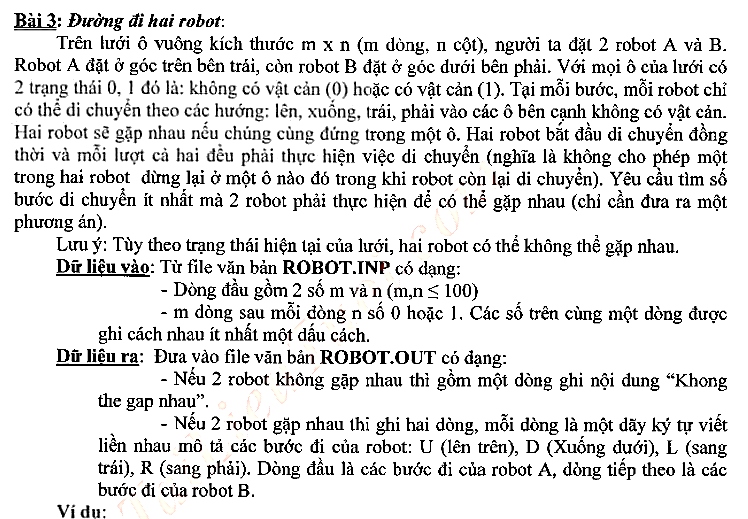


- Dữ liệu vào: Giá trị của N được cho trong tập tin **InB1.txt**

- Dữ liệu ra: Ghi trong tập tin **OutB1.txt** mảng kết quả

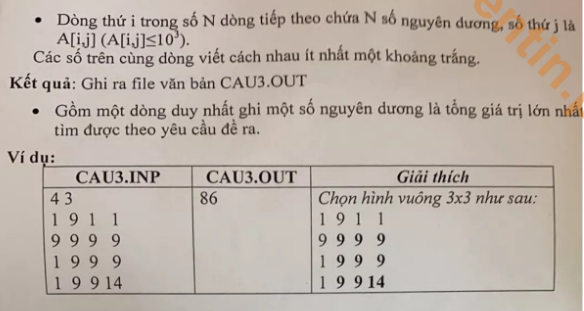
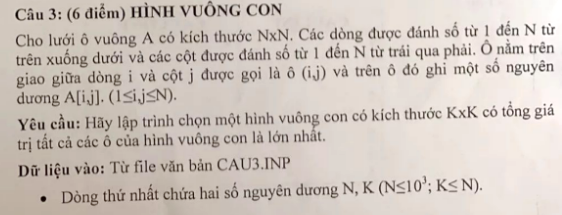


**Câu 4**

****

**Câu 5**

**Câu 6**



**Câu 7**

**7. Di chuyển**

**7.1. Mô hình**

Cho bảng A gồm M∗N ô. Từ ô (i,j)có thể di chuyển sang 3 ô (i+1,j), (i+1,j−1) và (i+1,j+1). Hãy xác định một lộ trình đi từ hàng 1 đến hàng M sao cho tổng các ô đi qua là lớn nhất.

**7.2. Công thức**

Gọi F(i,j) là giá trị lớn nhất có được khi di chuyển đến ô (i,j). Có 3 ô có thể đi đến ô (i,j) là (i−1,j), (i−1,j−1) và (i−1,j+1). Do đó ta có công thức QHĐ như sau:

* F[1,j]=A[1,j]   
  F[i,j]=max(F[i−1,j],F([i−1,j−1],F[i−1,j+1])+A[i,j] với i>1

**7.3. Cài đặt**

Bảng phương án là bảng 2 chiều F[0..m,0..n]. (*Tất cả các ô trên biên đều cho giá trị bằng 0).*

**Quá trình tính như sau:**

*for i:=1 to m do  
for j := 1 to n do*

*F[i,j]=max(F[i-1,j],F[i-1,j-1],F[i-1,j+1]]+A[i,j]);*

**Cách cài đặt này cho độ phức tạp bộ nhớ và thời gian đều là O(n2). Ta có thể tiết kiệm không gian nhớ bằng cách tính trực tiếp trên mảng A.**

*include<bits/stdc++.h>*

*using namespace std;*

*int i,j,m,n,a[1000][1000],b[1000][1000],ii[1000],jj[1000],k=0;*

*int main()*

*{*

*cin>>m>>n;*

*for(i=1;i<=m;i++)*

*for(j=1;j<=n;j++) cin>>a[i][j];*

*for(i=0;i<=m;i++) b[0][i]=0;*

*for(i=0;i<=m;i++) b[i][0]=0;*

*for(i=1;i<=m;i++)*

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

*b[i][j]=max(max(b[i-1][j],b[i-1][j-1]),b[i][j-1])+a[i][j];*

*for(i=1;i<=m;i++)*

*{*

*for(j=1;j<=n;j++) cout<<b[i][j]<<" ";*

*cout<<endl<<endl;*

*}*

*i=m;j=n; ii[1]=i;*

*jj[1]=j;*

*while(i>1||j>1)*

*{*

*k++;*

*ii[k]=i;*

*jj[k]=j;*

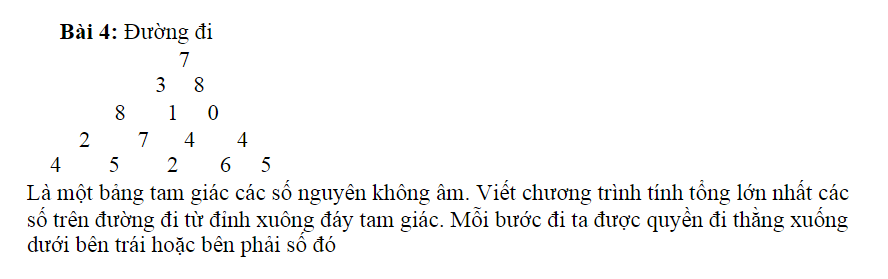
*if(b[i][j]==a[i][j]+b[i-1][j])i--; else if(b[i][j]==a[i][j]+b[i-1][j-1]){i--;j--;}else if(b[i][j]==a[i][j]+b[i][j-1])j--;*

*}*

*cout<<b[m][n]<<endl;*

*cout<<1<<" "<<1<<endl;*

*for(i=k;i>=1;i--) cout<<ii[i]<<" "<<jj[i]<<endl;*

*}*

*#include<bits/stdc++.h>*

*using namespace std;*

*int i,j,m,n,a[1000][1000],b[1000][1000],ii[1000],jj[1000],k=0;*

*int main()*

*{*

*cin>>n;*

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

*for(j=1;j<=i;j++) cin>>a[i][j];*

*//for(i=0;i<=m;i++) b[0][i]=0;*

*//for(i=0;i<=m;i++) b[i][0]=0;*

*b[1][0]=-1;*

*b[1][1]=a[1][1];*

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

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

*b[i][j]=max(b[i-1][j],b[i-1][j-1])+a[i][j];*

*int ans = b[n][1];*

*for(i=2;i<=n;i++) if(ans<b[n][i]) ans=b[n][i];*

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

*{*

*for(j=1;j<=i;j++) cout<<b[i][j]<<" ";*

*cout<<endl<<endl;*

*}*

*cout<<ans;*

*}*

**CHUYÊN ĐỀ 5. CHUYÊN ĐỀ XÂU**

**1. Nhập xâu**

Nhập 1 xâu có dấu cách: **getline(cin,s);**

Nhập nhiều xâu: **cin>>s**

2.Xuất xâu:

**cout<<S;**

**3. Các hàm trên xâu**

**S.size()** :Chiều dài xâu

**S.find(T**) : Tìm vị trí xuất hiện đầu tiên xâu T trong S nếu không trả -1

**S.rfind(T)** : Tìm vị trí xuất hiện cuối cùng xâu T trong S nếu không trả -1

**S.substr(vt,n):** Sao chép xâu s từ vị trí vt, n phần tử

**3. Các thủ tục trên xâu**

**S.c\_str():** trả về 1 mảng ký tự biểu diễn s

**S.erase(vt,n):** Xóa xâu S từ vị trí vt xóa n phần tử

**T.insert(vt,S):** Chèn xâu s vào t tại vị trí vt

**atoll(S.c\_str()):** Chuyển xâu s thành số.

**itoa(x,k,10) :** chuyển số thành xâu

int x; char k[33]

**KỸ THUẬT LOANG ĐỐI XỨNG DÀI NHẤT**

for(i=0;i<s.size();i++)

{

if(s[i]==s[i+1])

{

int\_fast64\_t j=0;

while(s[i-j]==s[i+1+j]&&i+j<s.size()&&i-j>=0)j++;

j\*=2;

if(ans<j)ans=j;

}

if(s[i]==s[i+2])

{

int\_fast64\_t j=0;

while(s[i-j]==s[i+2+j]&&i+j<s.size()&&i-j>=0)j++;

j=j\*2+1;

if(ans<j)ans=j;

}

}

**KỸ THUẬT LOANG ĐẾM SỐ XÂU CON ĐỐI XỨNG**

for(i=0;i<s.size();i++)

{

if(s[i]==s[i+1])

{

int\_fast64\_t j=0;

while(s[i-j]==s[i+1+j]&&i+j<s.size()&&i-j>=0){j++;ans++;}

}

if(s[i]==s[i+2])

{

int\_fast64\_t j=0;

while(s[i-j]==s[i+2+j]&&i+j<s.size()&&i-j>=0){j++;ans++;}

}

}

cout<<ans+s.size();

**LUYỆN ĐỀ**

**Bài 1**

Xâu S được gọi là xâu con chung của xâu S1 và xâu S2 nếu xâu S là một dãy các ký tự liên tiếp trong S1 và cũng là dãy các ký tự liên tiếp trong S2.

**Yêu cầu:** Cho hai xâu kí tự S1 và S2 (có không quá 255 ký tự). Hãy tìm một xâu con chung S dài nhất của hai xâu S1 và S2. Ví dụ: S1 = ’Ky thi học sinh gioi Tinh môn Tin hoc’, S2 = ’hoc sinh gioi mon Tin hoc’ thì S = ‘hoc sinh gioi '.

Dữ liệu vào từ file văn bản Bai2.inp:

• Dòng đầu tiên ghi xâu S1;

• Dòng thứ hai ghi xâu S2.

Kết quả ghi ra file văn bản Bai2.out: Chỉ một số duy nhất là độ dài của xâu con chung dài nhất S. (Nếu hai xâu S1, S2 không có kí tự nào chung thì ghi số 0).

**Bài 2. BẬC ĐỐI XỨNG**

Xâu đối xứng là xâu đọc từ trái qua phải giống như đọc từ phải qua trái, chẳng hạn các xâu ′abba ′, ′madam ′ là các xâu đối xứng.

Người ta định nghĩa bậc đối xứng của xâu S, kí hiệu deg(S), như sau:

- Nếu S không phải là xâu đối xứng hoặc S có độ dài bằng 1 thì bậc đối xứng của S bằng 0 (viết là: deg(S) = 0)

- Nếu là xâu đối xứng có độ dài lớn hơn 1, giả sử S = S1S2…SN (N > 1), thì bậc đối xứng của S được tính theo công thức: deg(S) = 1 + deg(T), trong đó xâu T là nửa trái của S theo nghĩa: T = S1S2 …Sk với k = (N + 1) div 2.

Chẳng hạn:  
- deg(′x') = 0 vì ′x′ có độ dài bằng 1  
- deg(′x0′) = 0 vì ′x0′ không đối xứng  
- deg(′x0x' ) = 1 vì deg(′x0x′) = 1 + deg(′x0′)  
- deg(′x0xx0x') = 2 vì deg('x0xx0x ′) = 1 + deg(′ x0x')

Cho xâu độ dài không vượt quá 200 chỉ gồm các chữ cái Latin in thường, hãy xác định bậc đối xứng của .

**Dữ liệu (PALINDEG.INP)**

- Dòng 1: xâu S

**Kết quả (PALINDEG.OUT)**

- Dòng 1: số nguyên là bậc đối xứng của xâu S.

**Bài 3 (4.0 điểm) Chữ cái xuất hiện**

Cho xâu St chỉ gồm các chữ cái. Tính số lần xuất hiện của chữ cái xuất hiện nhiều nhất trong xâu (không phân biệt chữ in hoa và in thường).

**Dữ liệu vào:**Từ file BAI3.INP gồm: Xâu St (độdài ≤ 500 ký tự).

**Kết quả:** Ghi ra file BAI3.OUT gồm: Một dòng duy nhất là bội số chung nhỏ nhất của kết quả bài toán và 105.

**Bài 4: (3,5 điểm) Tính tổng *TONG.PAS***

Cho hai số nguyên dương M và N, M có p chữ số và N có q chữ số.

**Yêu cầu:** Tính tổng của hai số M và N.

**Dữ liệu vào:** Cho trong file văn bản TONG.INP có cấu trúc như sau:

- Dòng 1: Ghi số nguyên dương p là số lượng chữ số của M (1 ≤ p ≤ 30000).

- Dòng 2: Ghi p chữ số của M theo thứ tự từ trái sang phải, các chữ số được ghi cách nhau ít nhất một dấu cách.

- Dòng 3: Ghi số nguyên dương q là số lượng chữ số của N (1 ≤ q ≤ 30000).

- Dòng 4: Ghi q chữ số của N theo thứ tự từ trái sang phải, các chữ số được ghi cách nhau ít nhất một dấu cách.

**Dữ liệu ra:** Ghi ra file văn bản TONG.OUT theo cấu trúc như sau:

- Dòng 1: Ghi số nguyên dương k là số lượng chữ số của tổng tìm được.

- Dòng 2: Ghi k chữ số của tổng tìm được theo thứ tự từ trái sang phải, các chữ số được ghi cách nhau ít nhất một dấu cách.

**Bài 5:** (3,5 điểm) **Dãy con chung dài nhất *DAYCON.PAS***

Cho dãy số nguyên A gồm N phần tử a1, a2, ..., aN và dãy số nguyên B gồm M phần tử b1, b2, ..., bM. Các phần tử trong một dãy số có giá trị khác nhau từng đôi một.

(1 ≤ ai, bj ≤ 2.109; 1 ≤ N ≤ 100; 1 ≤ i ≤ N; 1 ≤ M ≤ 100; 1 ≤ j ≤ M).

Dãy C được gọi là dãy con của dãy A nếu dãy C nhận được từ dãy A bằng cách xóa đi một số phần tử và giữ nguyên thứ tự của các phần tử còn lại.

Nếu dãy C là dãy con của dãy A và cũng là dãy con của dãy B thì dãy C được gọi là dãy con chung của hai dãy A và B.

**Yêu cầu:** Hãy tìm dãy C là dãy con chung của hai dãy A và B sao cho số lượng phần tử của dãy C là lớn nhất.

**Dữ liệu vào:** Cho trong file văn bản DAYCON.INP có cấu trúc như sau:

- Dòng 1: Ghi số nguyên dương N là số lượng phần tử của dãy A.

- Dòng 2: Ghi N số nguyên là giá trị của các phần tử trong dãy A, các số được ghi cách nhau ít nhất một dấu cách.

- Dòng 3: Ghi số nguyên dương M là số lượng phần tử của dãy B.

- Dòng 4: Ghi M số nguyên là giá trị của các phần tử trong dãy B, các số được ghi cách nhau ít nhất một dấu cách.

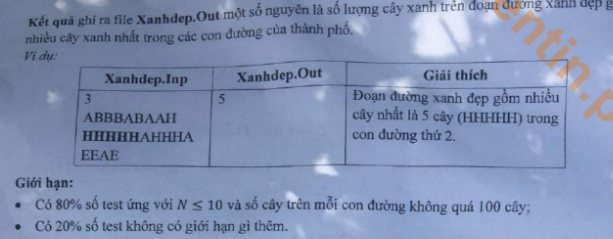
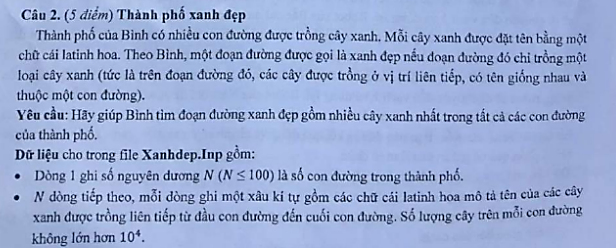
**Dữ liệu ra:** Ghi ra file văn bản DAYCON.OUT theo cấu trúc như sau:

- Dòng 1: Ghi số nguyên dương K là số lượng phần tử của dãy C.

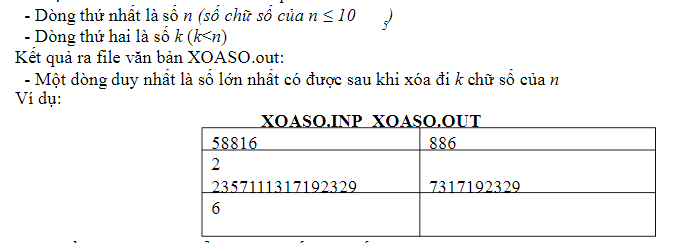
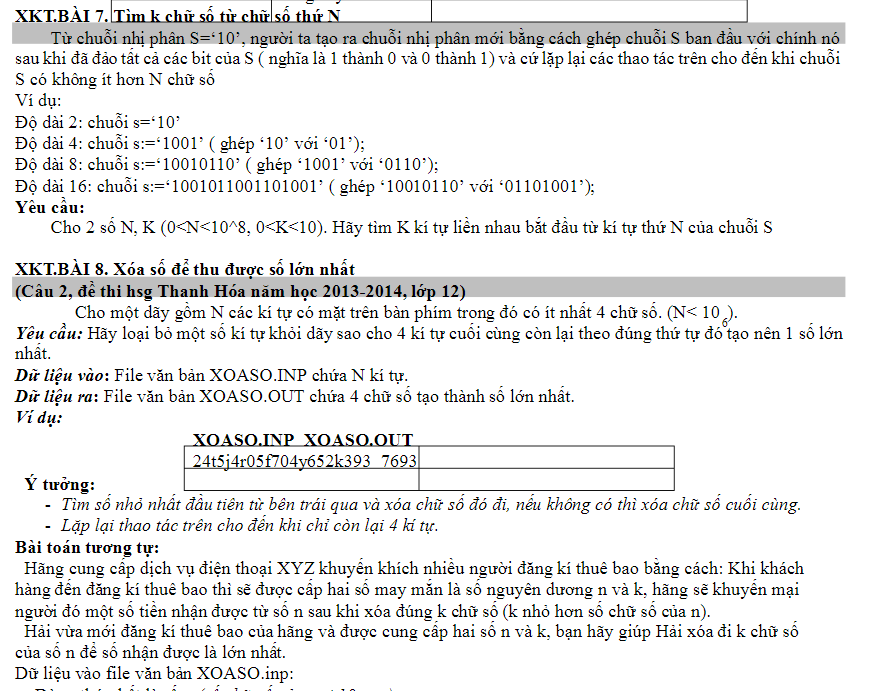
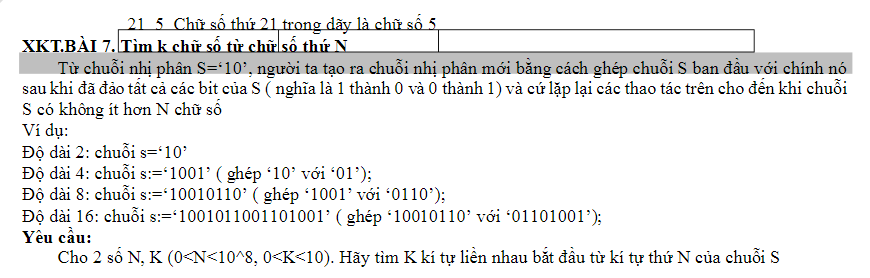
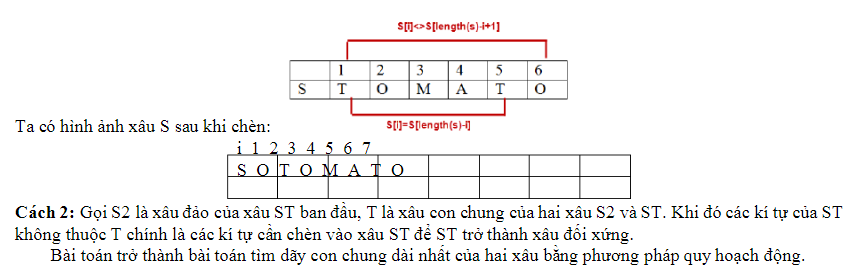
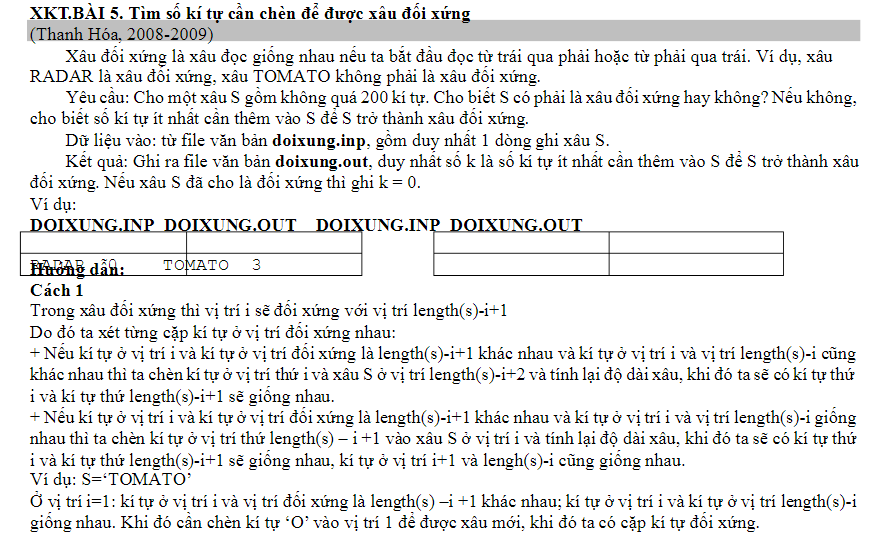
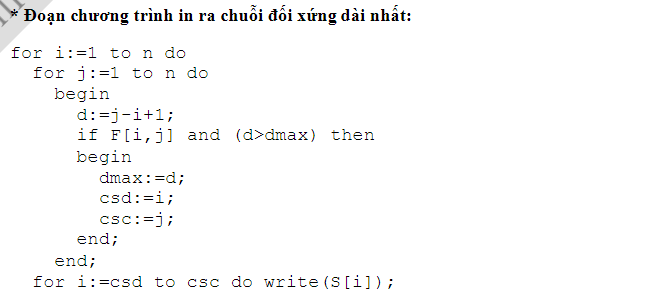
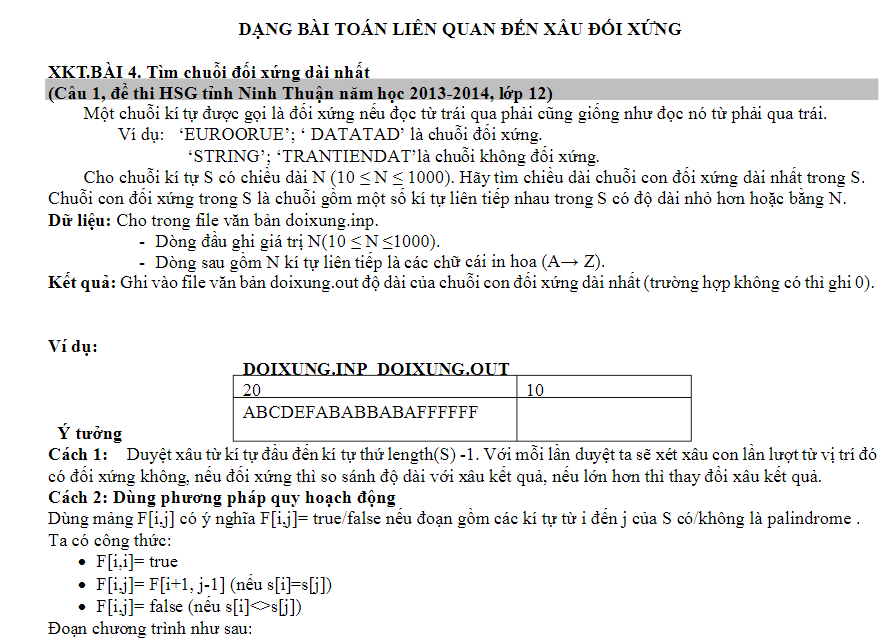
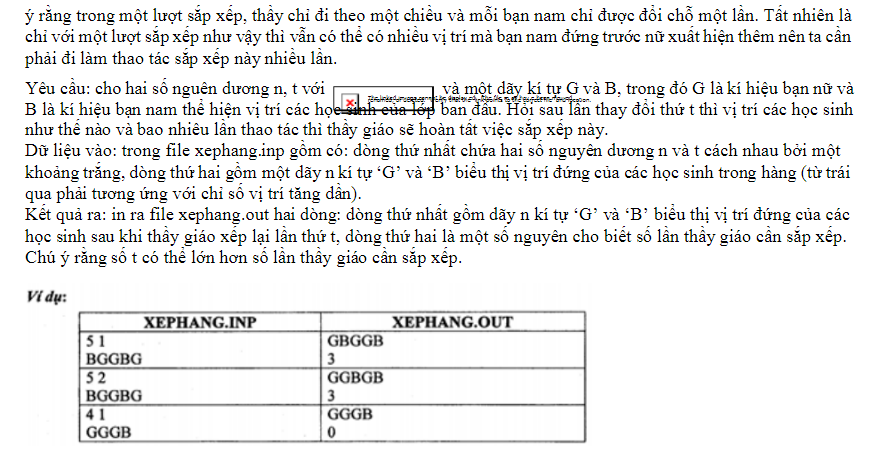
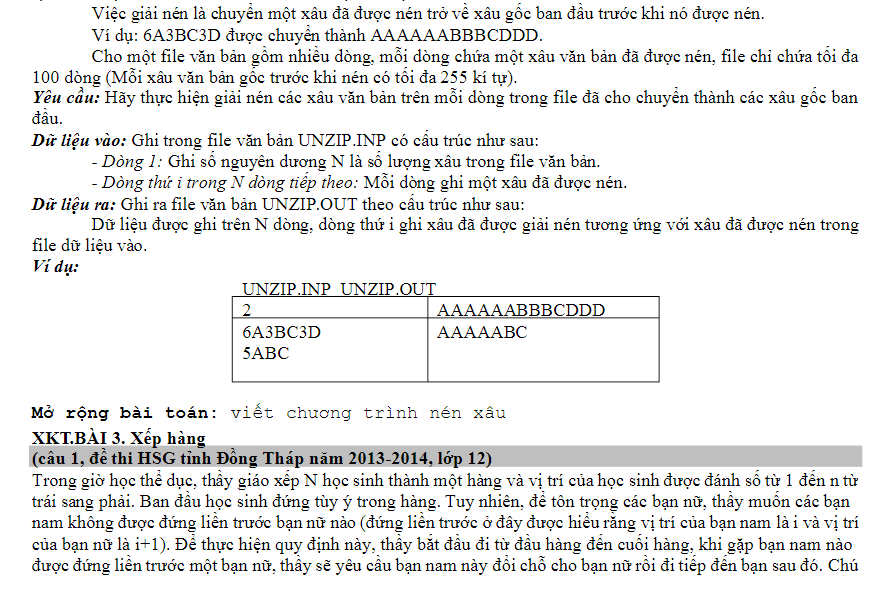
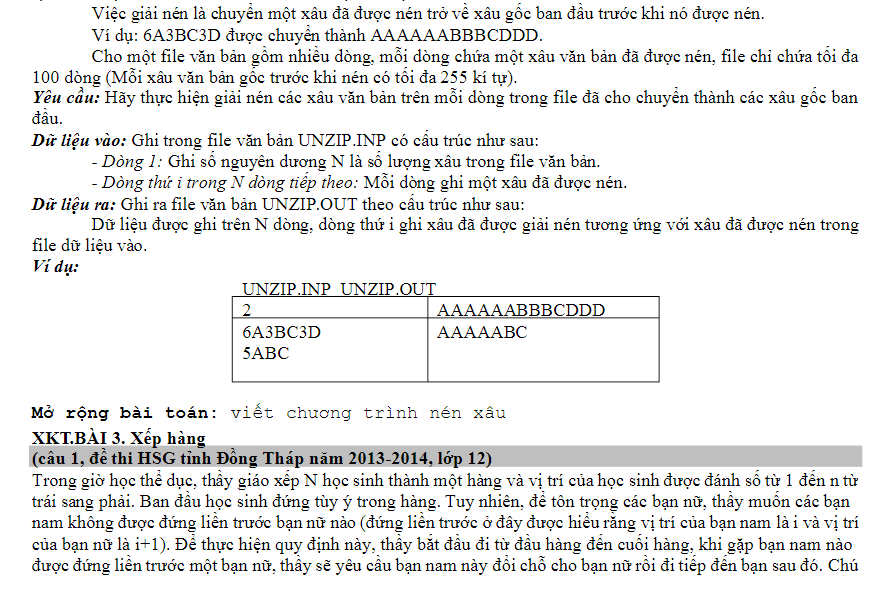
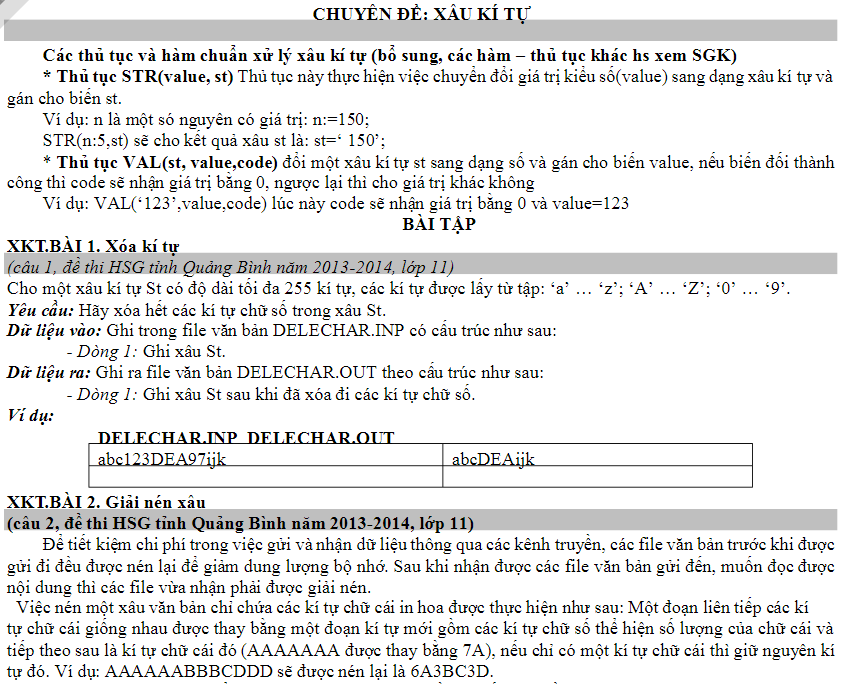
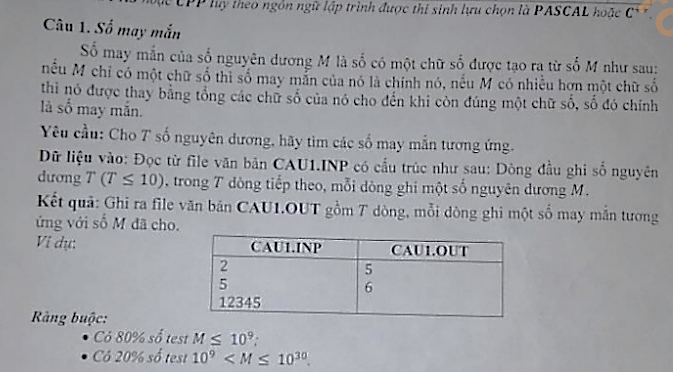
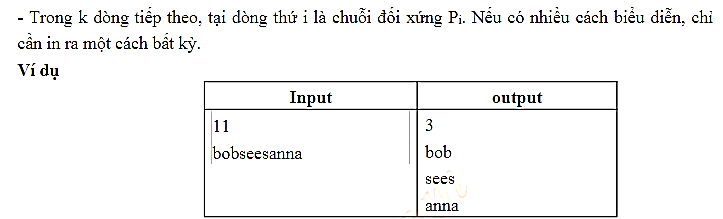
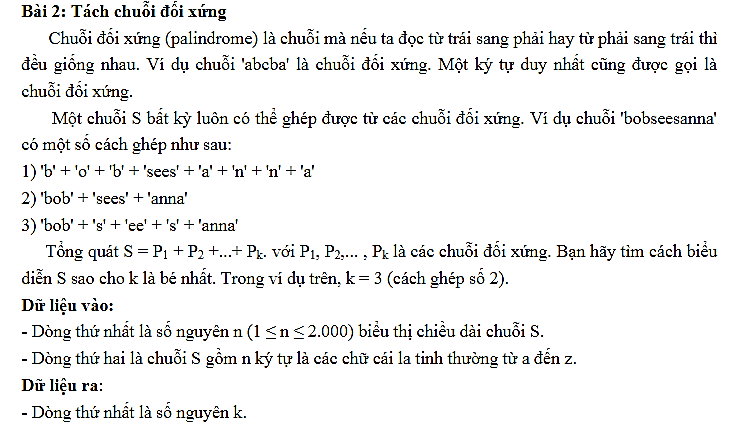
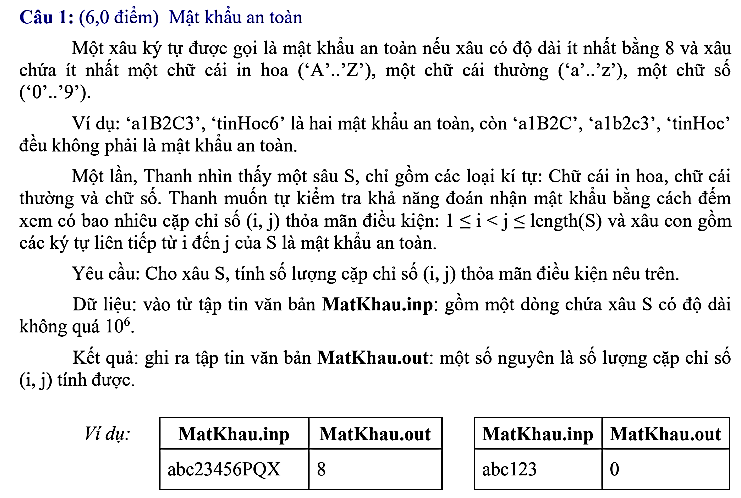
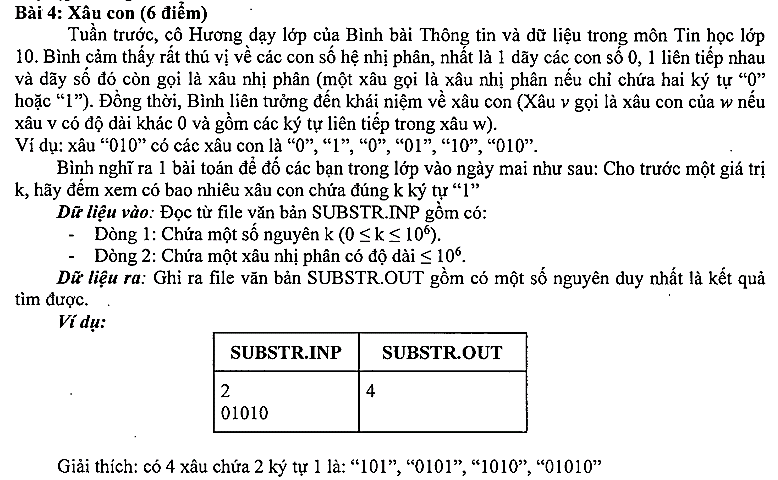
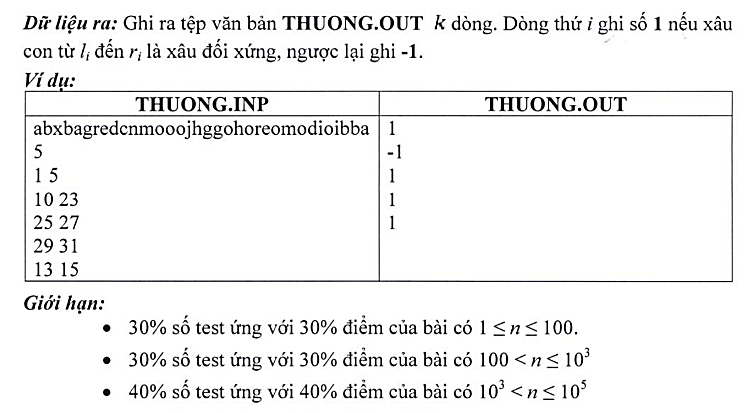
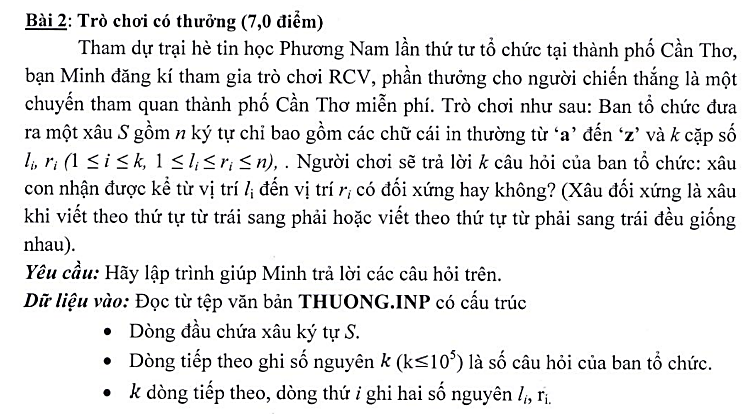
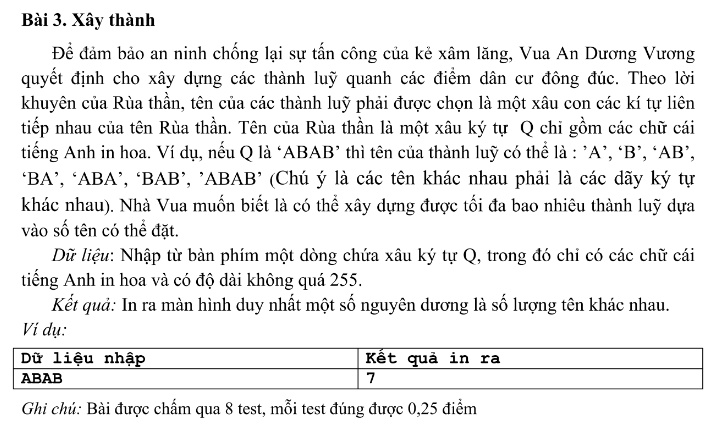
- Dòng 2: Ghi K số nguyên là giá trị của các phần tử trong dãy C, các số được ghi cách nhau một dấu cách.

- Dòng 3: Ghi K số nguyên dương lần lượt là chỉ số của các phần tử trong dãy A tương ứng với các giá trị của phần tử đó trong dãy C, các số được ghi cách nhau một dấu cách.

- Dòng 4: Ghi K số nguyên dương lần lượt là chỉ số của các phần tử trong dãy B tương ứng với các giá trị của phần tử đó trong dãy C, các số được ghi cách nhau một dấu cách.



**Bài 6**



**Bài 8**

**Bài 7**

**Bài 9**

**Bài 11**

**Bài 10**

**Bài 12**

**CHUYÊN ĐỀ 6.. MẢNG 1 CHIỀU**

**1. Nhập mảng**

cin>>n;

for( i=1;i<=n;i++) cin>>A[i];

**2. Xuất mảng**

for( i=1;i<=n;i++) cout<<A[i]<<“ “;

**3. Các hàm , thủ tục trên mảng**

**x=\*max\_element(a+1,a+1+n):** Giá trị lớn nhất của mảng a

**y=max\_element(a+1,a+1+n)-a:** vị trí của Giá trị lớn nhất trong mảng a

**x=\*min\_element(a+1,a+1+n):** Giá trị nhỏ nhất của mảng a

**y=min\_element(a+1,a+1+n)-a**: vị trí của Giá trị nhỏ nhất trong mảng a

**sort(a,a+n):** Sắp xếp a tăng dần

bool comp(int x,int y)

{

return (x>y);

}

**sort(a,a+n,comp):** Sắp xếp a giảm dần

**lower\_bound(a,a+n,val):** Vị trí đầu tiên lớn hơn hoặc bằng Val trong mảng a đã sx

**upper\_bound(a,a+n,val):** Vị trí đầu tiên lớn hơn Val trong mảng a đã sắp xếp

**LUYỆN ĐỀ**

**Bài 1** (7 điểm):

Một điểm giao dịch của ngân hàng X có N loại tiền mệnh giá từ A[1], A[2], A[3], . . , A[N] (đơn vị ngàn đồng) với số lượng tiền mỗi loại không giới hạn. Một khách hàng cần rút với số tiền là M (ngàn đồng). Hãy cho biết cần bao nhiêu tiền mỗi loại để chi trả sao cho số tờ là ít nhất.

Cho biết: N ≤ 9; A[i] ≤ 500; M ≤ 10000

**Dữ liệu vào:** Cho trong file INP.TXT gồm 2 dòng:

- Dòng đầu là 2 sốN, M;

- Dòng thứ hai ghi N số nguyên dương A[1], A[2], A[3], . . , A[N]

**Dữ liệu ra:** Ghi vào file OUT.TXT gồm:

- Dòng đầu ghi số lượng tờ phải trả;

- Dòng thứ hai ghi N số nguyên không âm ứng với số tờ cần trả cho mỗi loại tiền.

Các số ghi trên cùng một dòng được cách ít nhất một dấu cách.

**Bài 2. DÃY CON**

Cho dãy số nguyên A = (a1, a2, … , aN) và số nguyên dương M. Hãy tìm cách xóa bỏ trong dãy A một số phần tử sao cho dãy con thu được có nhiều phần tử nhất đồng thời trong dãy con này không có hai phần tử nào có tổng chia hết cho M.

Chẳng hạn, với N = 5, M = 3, A = (1, 2,3,4,5), dãy con dài nhất thu được có độ dài 3, có 4 dãy con như vậy, đó là (1,2,3); (1,3,4); (2,3,5); (3,4,5).

**Dữ liệu (CONFLICT.INP)**

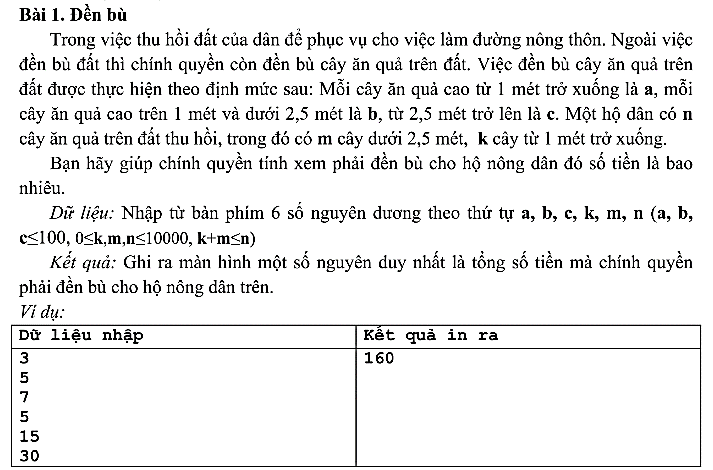
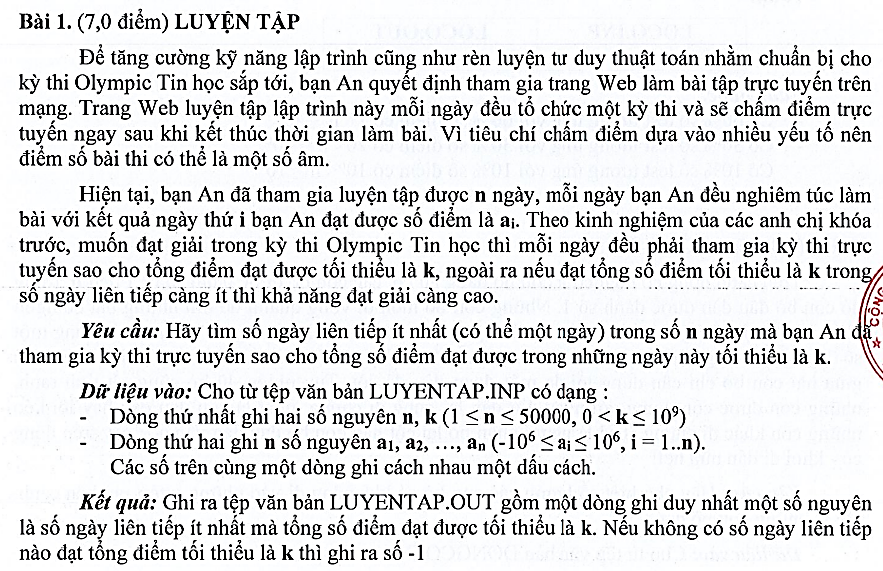
- Dòng 1: hai số nguyên N, M (1 ≤ N ≤ 10 ; 2 ≤ M ≤ 105)

- Dòng 2: số nguyên a1, a2, … , aN (|ai| ≤ 10 ∀i = 1 ÷ N).

**Kết quả (CONFLICT.OUT)**

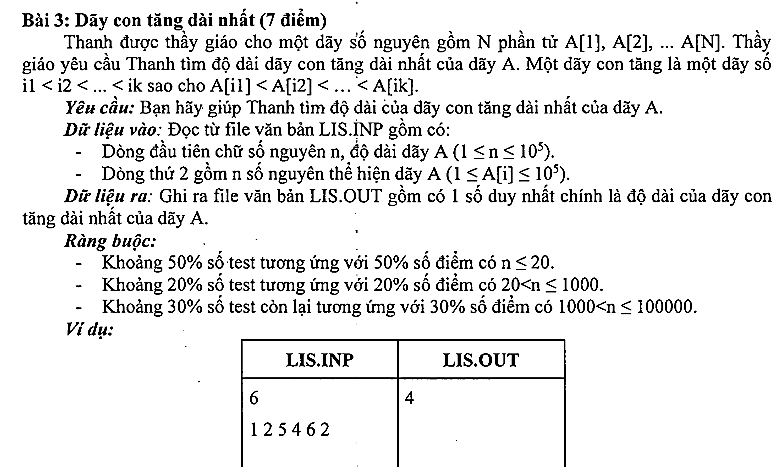
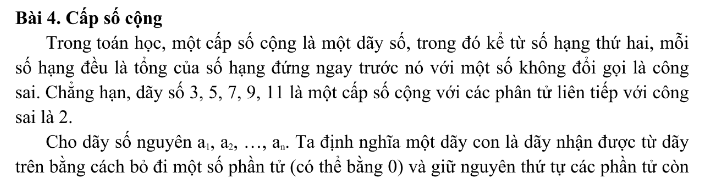
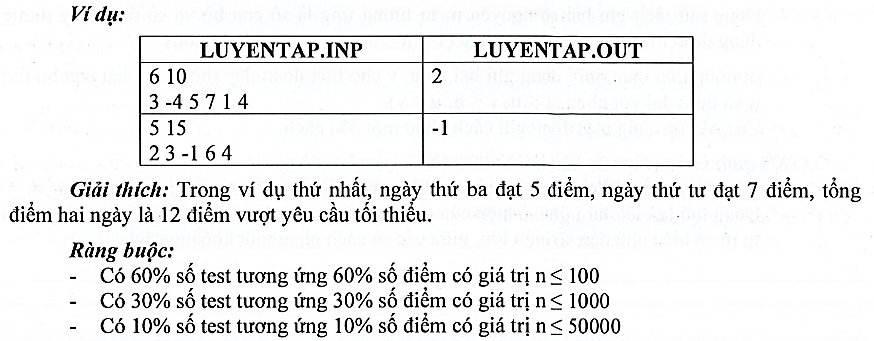
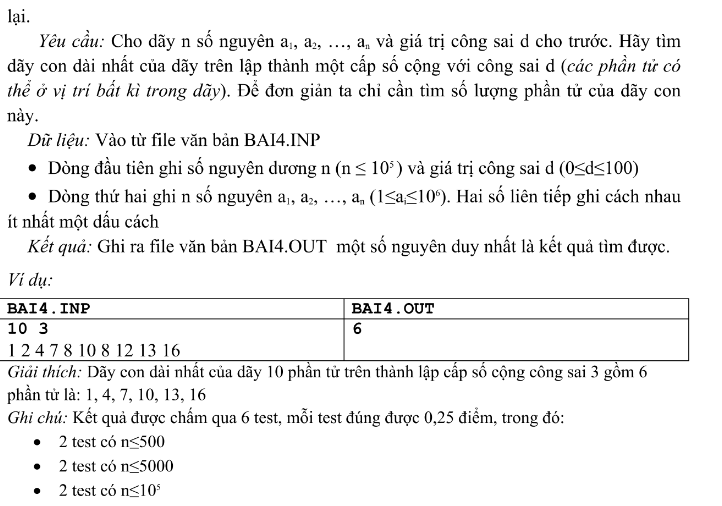
- Dòng 1: số nguyên K là số phần tử của dãy con thu được,

- Dòng 2: K số nguyên là chỉ số trong dãy ban đầu của các phần tử dãy con thu được, các số đưa ra theo trật tự tăng. Nếu có nhiều cách xóa cho dãy con độ dài K hỏa mãn yêu cầu bài toán thì chỉ cần đưa ra một cách.



**Bài 4**

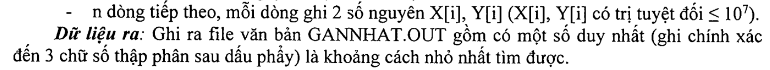
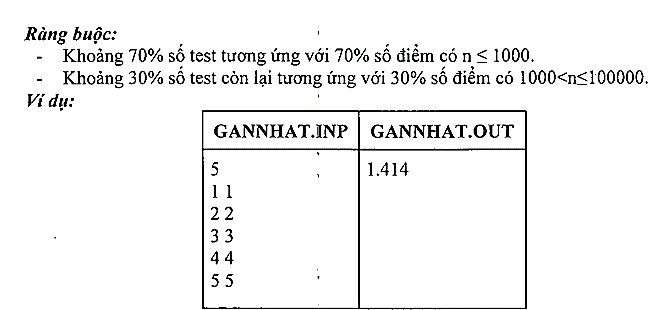
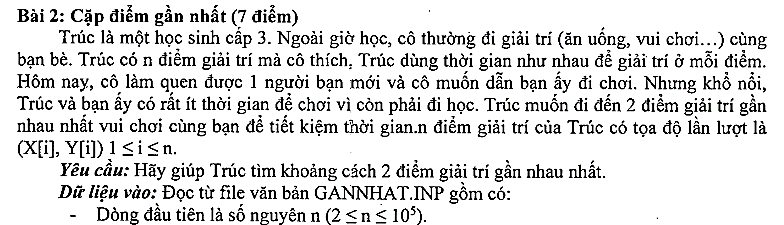
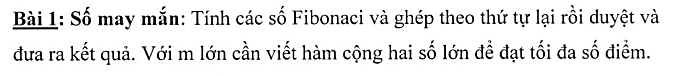
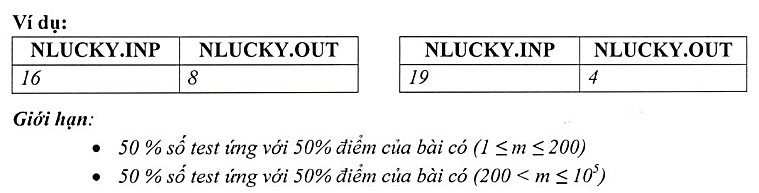
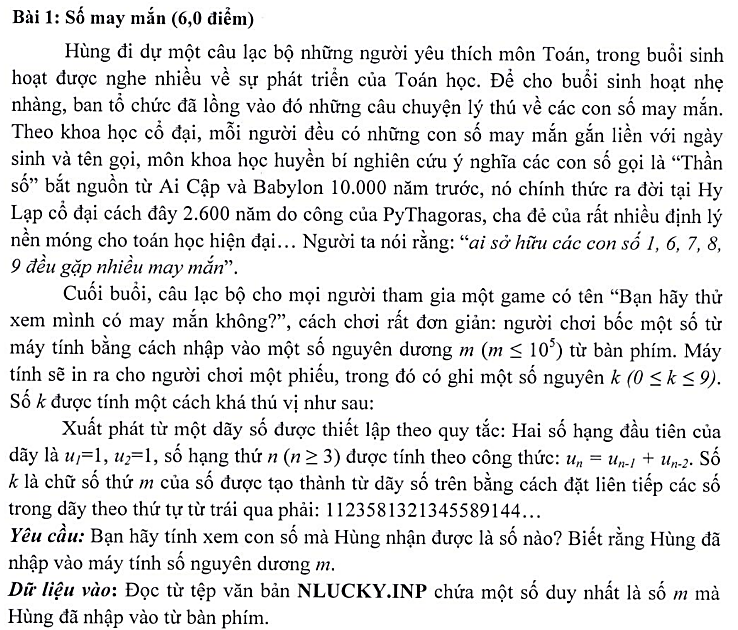
**Bài 3**



**Bài 5**

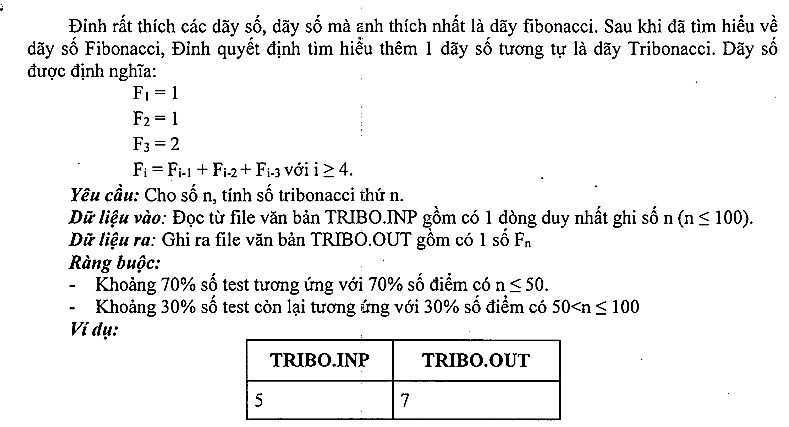
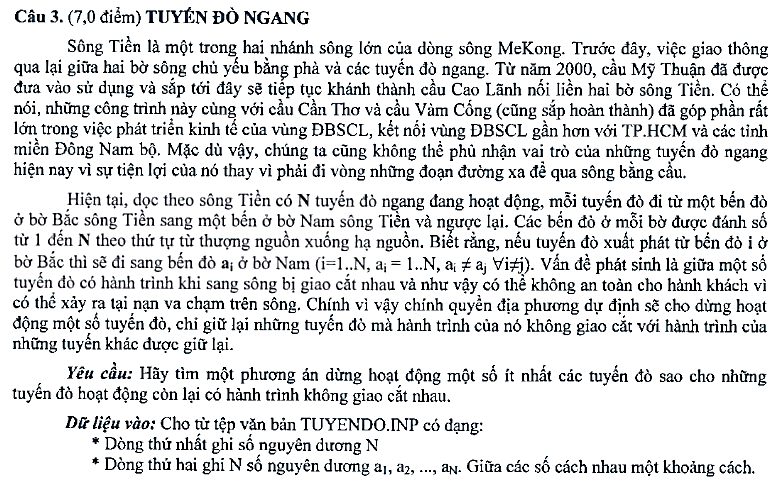
**Bài 7**

**Bài 6**



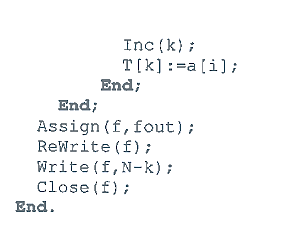
**HD**

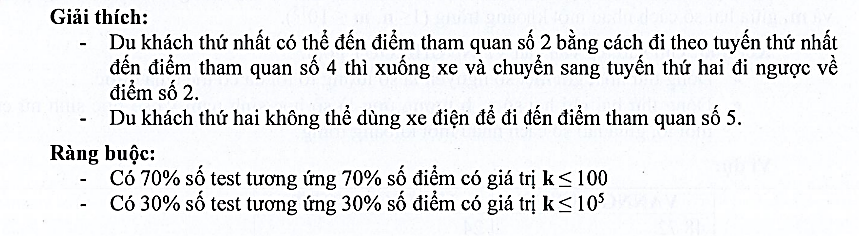
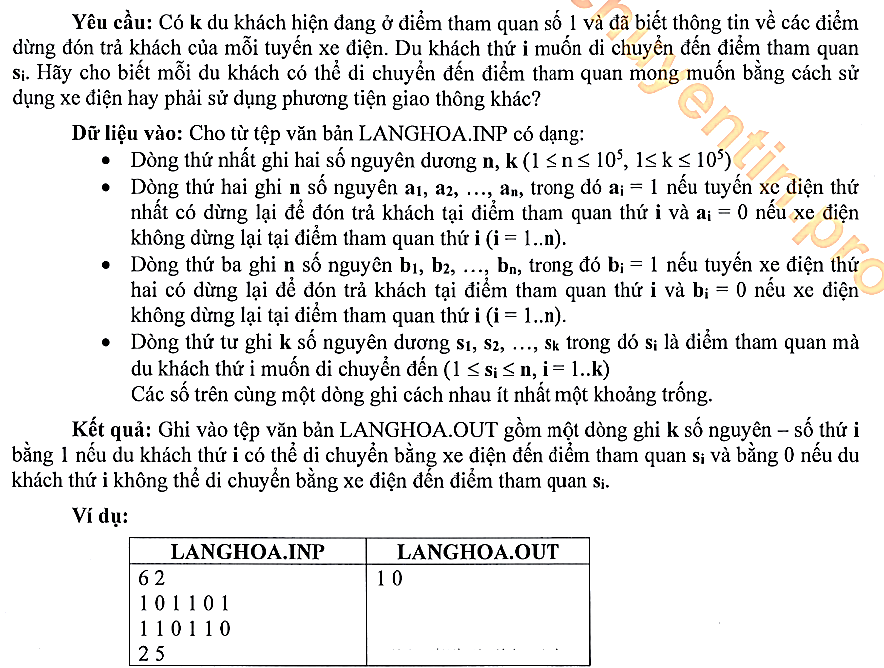
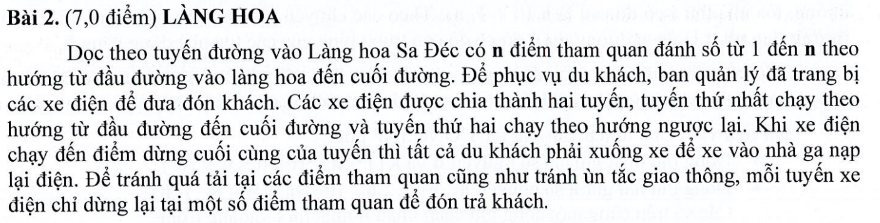
**Bài 8**



**Bài 10**

**Bài 9**





**Bài 11**