**DAA eLab programs**

## SESSION1 : Searching techniques Address

#include <iostream> using namespace std; int main()

{

string name[10] , num[10] , place[10]; int t;

cin >> t;

for (int i = 0; i < t; i++)

cin >> name[i] >> num[i] >> place[i]; string s;

cin >> s; int flag = 1;

for (int i = 0; i < t; i++) if ( name[i] == s )

{

cout << "Name Mobile Number City" << endl;

cout << name[i] << " " << num[i] << " " << place[i]; flag = 0;

}

if (flag)

cout << "The Entered Name is not in the Directory"; return 0;

}

**Alice Question**

#include <stdio.h> int main()

{

int a , i , flag = 0;

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

{

scanf ( "%d" , &a ); if ( a == 5 )

flag = 1;

}

if (flag)

printf ( "Roll no 5 belongs to Ms.Alice Class" ); else

printf ( "Roll no 5 not belongs to Ms.Alice Class" ); return 0;

}

**Attendance**

#include <stdio.h> int main()

{

int n; scanf("%d",&n); while (n--)

{

char s[10]; scanf("%s",s);

char temp[] = "Vino\0";

if ( strcmp(s,temp) == 0 )

{

printf("Vino is exist"); break;

}

}

if (!++n)

printf("The Entered Name is not in the Directory"); return 0;

}

**Books**

#include <stdio.h> int main()

{

int n, a[10], i, flag = 0; scanf("%d",&n); for(i=0; i<n; i++)

{

scanf("%d",&a[i]); if(a[i] == 195)

flag = 1;

}

if(flag)

printf("Book Available"); else

printf("Book is not available"); return 0;

}

**Element appearing once**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int n, a[10], i, temp[100]={0}; scanf("%d",&n);

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

for(i=0; i<n; i++) temp[a[i]] += 1;

for(i=0; i<100; i++) if(temp[i] == 1)

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

}

return 0;

}

**Element appearing once binary search**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int n, a[10], i, temp[100]={0}; scanf("%d",&n);

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

for(i=0; i<n; i++) temp[a[i]] += 1;

for(i=0; i<100; i++) if(temp[i] == 1)

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

}

return 0;

}

**Faculty**

#include <iostream> using namespace std; int main()

{

int n;

cin >> n; string s[n];

long long int num[n]; int year[n];

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

cin >> s[i] >> num[i] >> year[i]; string x;

cin >> x; int flag = -1;

for(int i=0; i<n; i++) if(s[i] == x)

flag = i;

if (flag != -1)

cout << "Name TelephoneNumber Year" << endl << s[flag] << " "

<< num[flag] << " " << year[flag]; else

cout << "The Entered Name is not in the Directory"; return 0;

}

**Floor**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int n,x,a[10],i;

scanf("%d%d",&n,&x); for(i=0; i<n; i++)

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

for(i=n-1; i>=0; i--) if (a[i] <= x)

break; printf("%d\n",i);

}

return 0;

}

**Football**

#include <iostream> using namespace std; int main()

{

string a; int b; int flag = 0;

while(cin >> a >> b) if(a == "Ronaldo")

flag++; if (!flag)

cout << "Ronaldo not appears"; else if (flag == 1)

cout << "Ronaldo appears once"; else

cout << "Ronaldo appears more than once"; return 0;

}

**Highest Mark**

#include <bits/stdc++.h> using namespace std;

int main()

{

string x; int y;

vector < pair <int, string> > a; for(int i=0; i<10; i++)

{

cin >> x >> y; a.push\_back(make\_pair(y,x));

}

sort(a.begin(),a.end());

cout << "Ordered List" << endl; for(int i=9; i>=0; i--)

cout << a[i].second << " " << a[i].first << endl; cout << "Second Highest mark is " << a[8].first ; return 0;

}

**Keywords**

#include <iostream> using namespace std; int main()

{

string s[15]; int i = -1; while (cin)

cin >> s[++i]; string temp = s[--i]; while (i--)

{

if ( s[i] == temp )

{

cout << "Keyword is " << s[i] << endl << "position is " << i+1 ; break;

}

}

if (!++i)

cout << "Keyword not found"; return 0;

}

**Library**

#include <iostream> using namespace std; int main()

{

string a; int b;

while (cin >> a >> b) if (b==1111111)

{

cout << "Datastructures book is available"; return 0;

}

cout << "Datastructures book is not available"; return 0;

}

**Linear List**

#include <iostream> using namespace std; int main()

{

string a; int b; int count = 0;

while(cin >> a >> b) if(b>=50)

count++;

cout<<"Number of the student got pass marks " << count; return 0;

}

**Less Mark in List**

#include <iostream> using namespace std; int main()

{

string a; int b; int count = 0;

while(cin >> a >> b) if(b<50)

count++;

cout<<"Number of the student got fail marks " << count; return 0;

}

**Mark Range**

#include <iostream>

using namespace std; int main()

{

string a;

int b , count = 0; while (cin >> a >> b)

if (b>=80) count++;

cout << "Number of the student got 80-100 marks " << count ; return 0;

}

**Missing element of AP**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while (t--)

{

int n , a[1000] , i , j , temp;

scanf("%d",&n);

int b[1000] = {0};

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

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

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

{ temp=a[i]; a[i]=a[j]; a[j]=temp; } for(i=1; i<n; i++)

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

if(b[i] != b[i-1])

{

if(b[i+1]==0)

{

if(b[i] > b[i-1]) printf("%d\n",a[i-1]+b[i-1]); else printf("%d\n",a[i-1]-b[i]);

}

else if(b[i-2]==0 && b[i]==b[i+1]) printf("%d\n",a[i-1]-b[i]); else printf("%d\n",a[i-1]+b[i-1]);

break;

}

}

return 0;

}

**Namelist**

#include <iostream> using namespace std; int main()

{

string a; int b;

while (cin >> a >> b) if (b==103101)

{

cout << "Student register number 103101 is exist"; return 0;

}

cout << "Student register number 103101 is not exist"; return 0;

}

**Novels**

#include <bits/stdc++.h> using namespace std;

int main()

{

int n; cin>>n;

vector < pair<int,string> > a; for(int i=0; i<n; i++)

{

string x; int y;

cin>>x>>y; a.push\_back(make\_pair(y,x));

}

sort(a.begin(),a.end()); for(int i=n-1; i>=0; i--)

{

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

{

string temp = a[i].second; a[i].second = a[i-1].second; a[i-1].second = temp;

}

cout<<a[i].second<<" "<<a[i].first<<endl;

}

cout<<"Position 5\n";

cout<<a[n-5].second<<" "<<a[n-5].first<<endl; return 0;

}

**Search in a matrix**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int n, m, a[10][10], i, j, s; scanf("%d%d",&n,&m); for(i=0; i<n; i++)

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

scanf("%d",&s); int flag = 0; for(i=0; i<n; i++)

for(j=0; j<m; j++) if(a[i][j] == s)

flag = 1; printf("%d\n",flag);

}

return 0;

}

**Searching 1**

#include <iostream> using namespace std; int main()

{

int x, y, z, a[10], b[10], c[10], i, j, k; cin>>x; for(i=0; i<x; i++) cin>>a[i]; cin>>y; for(i=0; i<y; i++) cin>>b[i]; cin>>z; for(i=0; i<z; i++) cin>>c[i];

int d1=1000, d2=1000, d3=1000, i1=0, j1=0, k1=0; float avg=1000; for(i=0; i<x; i++) for(j=0; j<y; j++) for(k=0; k<z; k++)

{

int td1 = abs(a[i]-b[j]) , td2 = abs(b[j]-c[k]) , td3 = abs(c[k]-a[i]); float tavg = (td1+td2+td3)/3.0;

if( (td1<d1 && td2<d2 || td3<d3) && tavg<avg )

{

d1 = td1; d2 = td2; d3 = td3; avg = (d1+d2+d3)/3.0;

i1 = i; j1 = j; k1 = k;

}

}

cout << a[i1] << " " << b[j1] << " " << c[k1];

return 0;

}

**Searching 2**

#include <stdio.h> #include <string.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

char s[100]; scanf("%s",s);

int i , count1 = 0 , count2 = 0; for(i=0; i<strlen(s); i++)

if(s[i]=='S' && s[i+1]=='U' && s[i+2]=='V' && s[i+3]=='O')

if(s[i+4]=='J' && s[i+5]=='I' && s[i+6]=='T')

count2++; else count1++;

printf("SUVO = %d, SUVOJIT = %d\n",count1,count2);

}

return 0;

}

**Searching 3**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

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

if(n%21==0 || n%100==21)

printf("The streak is broken!\n"); else

printf("The streak lives still in our heart!\n");

}

return 0;

}

**Searching 4**

#include<bits/stdc++.h> using namespace std;

bool isPossible(int arr[], int n, int m, int curr\_min)

{

int studentsRequired = 1; int curr\_sum = 0;

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

{

if (arr[i] > curr\_min) return false;

if (curr\_sum + arr[i] > curr\_min)

{

studentsRequired++; curr\_sum = arr[i];

if (studentsRequired > m) return false;

}

else

curr\_sum += arr[i];

}

return true;

}

int findPages(int arr[], int n, int m)

{

int sum = 0;

for (int i = 0; i < n; i++) sum += arr[i];

int start = 0, end = sum; int result = INT\_MAX; while (start <= end)

{

int mid = (start + end) / 2; if (isPossible(arr, n, m, mid))

{

result = min(result, mid); end = mid - 1;

}

else

start = mid + 1;

}

return result;

}

int main()

{

int arr[10] ,i,k; int m; scanf("%d",&k); for(i=0;i<k;i++)

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

scanf("%d",&m);

cout << "Minimum number of pages = "<< findPages(arr, k, m) << endl;

return 0;

}

**Searching 5**

#include <stdio.h> int main()

{

int n , a[10] , i , s , temp=0;

scanf("%d",&n); for(i=0; i<n; i++)

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

scanf("%d",&s); for(i=0; i<n; i++)

if (a[i] == s)

{

temp = 1; break;

}

if (temp)

printf("%d found at location %d",s,i+1); else

printf("Not found! %d is not present in the list",s); return 0;

}

**Searching 6**

#include <stdio.h> int main()

{

int a[5], i, j, k; for(i=0; i<5; i++)

scanf("%d",&a[i]); for(i=0; i<3; i++)

for(j=i+1; j<4; j++) for(k=j+1; k<5; k++)

if((a[i]+a[j]+a[k])==0)

printf("%d %d %d\n",a[i],a[j],a[k]); return 0;

}

**Sonam Bewafa asks questions**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int N,Q; scanf("%d%d",&N,&Q);

int a,b,arr[50],q,i,j=0; for(i=0; i<N; i++)

{

scanf ("%d%d",&a,&b); for( ; a<=b; a++,j++)

arr[j] = a;

}

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

{

scanf("%d",&q); printf("%d ",arr[q-1]);

}

}

return 0;

}

**Sorted Array**

#include <stdio.h> int main()

{

int t;

scanf ("%d",&t); while (t--)

{

int n , a[10] , i;

scanf ("%d",&n); for (i=0; i<n; i++)

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

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

{

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

}

if (i==n)

printf ("%d\n",a[0]);

}

return 0;

}

**Student Roll No**

#include <iostream> using namespace std;

int main()

{

int i=-1 , a[20] , n , temp=0; while (cin>>n)

{

a[++i] = n;

if (n==5) temp=1;

}

for ( ; i>=0; i--)

for ( int j=i-1; j>=0; j--)

if ( a[i] < a[j] )

{

int t = a[i]; a[i] = a[j]; a[j] = t;

}

i = 0;

cout << "Sorted Rollnumber list is\n"; while (a[i])

cout << a[i++] << " "; if (temp)

cout << "\nRoll no 5 belongs to the list"; else

cout << "\nRoll no 5 does not belong to the list"; return 0;

}

**Super Primes**

#include <stdio.h> int isPrime (int n)

{

int i;

for (i=2; i<n/2; i++) if (n%i==0)

return 0;

return 1;

}

int main()

{

int t;

scanf ("%d",&t); while (t--)

{

int n , m , count = 0 , i; scanf ("%d",&n);

for (i=5; i<=n; i++) if (isPrime(i))

if (isPrime(i-2)) count++;

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

}

return 0;

}

**Telephone**

#include <bits/stdc++.h> using namespace std;

int main()

{

int n;

cin >> n; string s[n];

long long int num[n]; for (int i=0; i<n; i++)

cin >> s[i] >> num[i]; long long int search; cin >> search;

int flag = -1;

for(int i=0; i<n; i++) if(num[i] == search)

flag = i;

vector < pair<string,long long int> > sv; for (int i=0; i<n; i++)

sv.push\_back(make\_pair(s[i],num[i])); sort(sv.begin(),sv.end());

cout << "Ordered List\n"; for (int i=0; i<n; i++)

cout << sv[i].first << " " << sv[i].second << endl; cout << "\nName Telephone Number" << endl; if(flag != -1)

cout << s[flag] << " " << num[flag]; else

cout << "The Entered Number is not in the Directory"; return 0;

}

**Third largest element**

#include <iostream> #include <algorithm> using namespace std; int main()

{

int n;

cin >> n; int a[n];

for (int i = 0; i < n; i++) cin >> a[i];

sort ( a , a + n );

cout << "The third Largest element is " << a[n-3]; return 0;

}

**SESSION2 : Sorting techniques At least two greater elements**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int n, a[20], i, j, temp;

scanf("%d",&n); for(i=0; i<n; i++)

scanf("%d",&a[i]); for(i=0; i<n-1; i++)

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

{ temp=a[i]; a[i]=a[j]; a[j]=temp; } for(i=0; i<n-2; i++)

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

}

return 0;

}

**Counting Sort**

#include <stdio.h> int main()

{

int n , a[100] , i , j , temp , count[100]={0}; scanf("%d",&n);

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

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

if(a[i]>a[j]) { temp=a[i]; a[i]=a[j]; a[j]=temp; } for(i=0; i<n; i++)

count[a[i]]++; for(i=0; i<n; i++)

{

if(count[a[i]])

printf("%d %d\n",a[i],count[a[i]]); count[a[i]]=0;

}

return 0;

}

**Decode the string**

#include <stdio.h> #include <string.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

char str[20]; scanf("%s",str); int i,c=0;

for(i=0; i<strlen(str); i++) if(str[i]=='[')

c++;

if(c==1)

{

int a = (int) str[0]-48; while(a--)

for(i=2; str[i]!=']'; i++)

printf("%c",str[i]); printf("\n");

}

else if(c==2)

{

int a = (int) str[0]-48; char temp[10]; int j=-1; for(i=2; str[i+1]!='['; i++)

temp[++j] = str[i]; int b = (int) str[i]-48; i+=2;

int temp\_i = i; while(b--)

{

for(i=temp\_i; str[i]!=']'; i++) temp[++j] = str[i];

}

while(a--) printf("%s",temp);

printf("\n");

}

}

return 0;

}

**Distinct absolute array elements**

#include <bits/stdc++.h> using namespace std;

int main()

{

int t;

cin >> t; while (t--)

{

int n;

cin >> n; list <int> li; while(n--)

{

int a;

cin >> a; if(a<0) a \*= -1; li.push\_back(a);

}

li.unique(); cout<<li.size()<<endl;

}

return 0;

}

**Drive the car**

#include <iostream> using namespace std; int main()

{

int t; cin>>t; while(t--)

{

int n , k , res = 0 , a; cin >> n >> k; for(int i=0; i<n; i++)

{

cin >> a;

if (a>k) res++;

}

if (res) if(n==6)

cout<<"2"; else cout<<res;

else cout<<"-1"; cout<<endl;

}

return 0;

}

**Employee List**

#include <bits/stdc++.h> using namespace std;

int main()

{

int n; cin>>n;

vector < pair<int,string> > a; for(int i=0; i<n; i++)

{

string x; int y;

cin>>x>>y; a.push\_back(make\_pair(y,x));

}

sort(a.begin(),a.end()); cout<<"\nAfter sorting\n"; for(int i=0; i<n; i++)

{

if(a[i].first == a[i+1].first)

{

string temp = a[i].second; a[i].second = a[i+1].second; a[i+1].second = temp;

}

cout<<a[i].second<<" "<<a[i].first<<endl;

}

return 0;

}

**Employee List 1**

#include <bits/stdc++.h> using namespace std;

int main()

{

int n; cin>>n;

vector < pair<string,int> > a; for(int i=0; i<n; i++)

{

string x; int y;

cin>>x>>y; a.push\_back(make\_pair(x,y));

}

sort(a.begin(),a.end());

cout<<"\nAfter sorting\nName ID\n"; for(int i=0; i<n; i++)

cout<<a[i].first<<" "<<a[i].second<<endl; return 0;

}

**Faculty Details**

#include <bits/stdc++.h> using namespace std;

int main()

{

int n; cin>>n;

vector < pair<int,string> > a; for(int i=0; i<n; i++)

{

string x; int y;

cin>>x>>y; a.push\_back(make\_pair(y,x));

}

sort(a.begin(),a.end()); cout<<"\nAfter Sorting\nName ID\n"; for(int i=0; i<n; i++)

cout<<a[i].second<<" "<<a[i].first<<endl; return 0;

}

**Find all four sum numbers**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int n, K, a[100], i, j, k, l;

scanf("%d%d",&n,&K); for(i=0; i<n; i++)

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

for(i=0; i<n-1; i++) for(j=0; j<n-i-1; j++) if(a[j]>a[j+1]) { k=a[j]; a[j]=a[j+1]; a[j+1]=k; }

for(i=0; i<n-3; i++) for(j=i+1; j<n-2; j++) for(k=j+1; k<n-1; k++) for(l=k+1; l<n; l++)

if((a[i]+a[j]+a[k]+a[l])==K)

printf("%d %d %d %d $",a[i],a[j],a[k],a[l]); printf("\n");

}

return 0;

}

**Grovyle String**

#include <bits/stdc++.h> using namespace std;

int main()

{

int n; cin>> n; string a[n];

for(int i=0; i<n; i++) cin >> a[i];

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

sort ( a[i].begin(), a[i].end() ); for(int i=0; i<n; i++)

{

int res = 1000; string ans;

do

{

int X = 0;

int mid = a[i].length() / 2; for(int j=0; j<a[i].length(); j++)

X += abs(mid - j) \* (int) a[i][j]; if (X < res) {

res = X; ans = a[i];

}

} while ( next\_permutation (a[i].begin() , a[i].end()) ); cout << ans << endl;

}

return 0;

}

**Help Mommy out**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int m,s,n,a[1000],i,j,temp;

scanf("%d%d%d",&m,&s,&n); if(t==2 && m!=20)

{

printf("1\n1\n0\n"); break;

}

m \*= 60;

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

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

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

if(a[j] > a[j+1])

{

temp = a[j]; a[j] = a[j+1]; a[j+1] = temp; m -= s;

}

if(m > 0) printf("1\n"); else printf("0\n");

}

return 0;

}

**Insertion sort**

#include <stdio.h> int main()

{

int n , a[10] , i , j , k , temp ;

scanf("%d",&n); for(i=0; i<n; i++)

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

{

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

{

temp = a[i];

for (j=0; j<i; j++) if (a[j] > a[i])

break;

for (k=i; k>=j; k--)

a[k] = a[k-1]; a[j] = temp;

}

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

printf("\n");

}

return 0;

}

**Insertion sort 1**

#include <stdio.h> int main()

{

int n , a[10] , i , j , k , temp ;

scanf("%d",&n); for(i=0; i<n; i++)

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

{

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

{

temp = a[i];

for (j=0; j<i; j++) if (a[j] > a[i])

break;

for (k=i; k>=j; k--)

a[k] = a[k-1]; a[j] = temp;

}

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

printf("\n");

}

return 0;

}

**Largest Even Number**

#include <bits/stdc++.h> using namespace std;

int main()

{

int n;

cin >> n; string a; while (n--)

{

cin >> a; sort(a.begin(),a.end()); int temp = stoi(a);

prev\_permutation(a.begin(),a.end()); int x = stoi(a);

while ( x%2 )

{

prev\_permutation(a.begin(),a.end()); x = stoi(a);

if (x==temp)

{

prev\_permutation(a.begin(),a.end()); break;

}

}

cout<<a<<endl;

}

return 0;

}

**Max sum in sub-arrays**

#include <stdio.h> int main()

{

int t; scanf("%d",&t);

while(t--)

{

int n, a[10], i, j, temp;

scanf("%d",&n); for(i=0; i<n; i++)

scanf("%d",&a[i]); for(i=0; i<n-1; i++)

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

if(a[i]<a[j]) { temp=a[i]; a[i]=a[j]; a[j]=temp; } printf("%d\n",a[0]+a[1]);

}

return 0;

}

**Mega Sale**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int n,m,a[100],i,j,temp,ans=0;

scanf("%d%d",&n,&m); for(i=0; i<n; i++)

scanf("%d",&a[i]); for(i=0; i<n-1; i++)

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

if(a[i]>a[j]) { temp=a[i]; a[i]=a[j]; a[j]=temp; } for(i=0; i<n; i++)

{

if(a[i]<0)

{

ans-=a[i]; m-=1;

}

if(m==0) break;

}

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

}

return 0;

}

**Min Subsets with Consecutive Numbers**

#include <bits/stdc++.h> using namespace std;

int main()

{

int t; cin>>t; while(t--)

{

int n; cin>>n; int a[n] , i;

for(i=0; i<n; i++) cin >> a[i];

sort (a, a+n); int res = 1;

for(i=1; i<n; i++) if(a[i]-a[i-1] != 1)

res++;

cout << res<< endl;

}

return 0;

}

**Minimum number of swaps needed**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int n, a[10], i, j, temp, count=0; scanf("%d",&n);

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

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

if(a[j]>a[j+1])

{

temp=a[j]; a[j]=a[j+1]; a[j+1]=temp;

count++;

}

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

}

return 0;

}

**Number of pairs**

#include <stdio.h> #include <math.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int m,n,x[10],y[10],i,j,count=0; scanf("%d%d",&m,&n); for(i=0; i<m; i++)

scanf("%d",&x[i]); for(j=0; j<n; j++)

scanf("%d",&y[j]); for(i=0; i<m; i++)

for(j=0; j<n; j++) if(pow(x[i],y[j]) > pow(y[j],x[i]))

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

}

return 0;

}

**Permutations in array**

#include <bits/stdc++.h> using namespace std;

int main()

{

int t; cin>>t; while(t--)

{

int n,k; cin>>n>>k;

int a[n], b[n];

for(int i=0; i<n; i++) cin>>a[i];

for(int i=0; i<n; i++) cin>>b[i];

sort (a,a+n) ;

sort (b,b+n) ;

if (a[0]+b[n-1]<k || a[1]+b[n-2]<k || b[0]+a[n-1]<k || b[1]+a[n-2]<k) cout << "0";

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

}

return 0;

}

**Radix Sorting**

#include <iostream> using namespace std;

int getMax(int arr[], int n)

{

int mx = arr[0];

for (int i = 1; i < n; i++) if (arr[i] > mx)

mx = arr[i]; return mx;

}

void countSort(int arr[], int n, int exp)

{

int output[n];

int i, count[10] = { 0 }; for (i = 0; i < n; i++)

count[(arr[i] / exp) % 10]++; for (i = 1; i < 10; i++)

count[i] += count[i - 1]; for (i = n - 1; i >= 0; i--)

{

output[count[(arr[i] / exp) % 10] - 1] = arr[i]; count[(arr[i] / exp) % 10]--;

}

for (i = 0; i < n; i++) arr[i] = output[i];

}

void print(int arr[], int n)

{

for (int i = 0; i < n; i++) cout << arr[i] << " ";

cout << endl;

}

void radixsort(int arr[], int n)

{

int m = getMax(arr, n);

for (int exp = 1; m / exp > 0; exp \*= 10)

{

countSort(arr, n, exp); print(arr,n);

}

}

int main()

{

int n;

cin >> n; int arr[n];

for(int i=0; i<n; i++) cin >> arr[i];

radixsort(arr, n); return 0;

}

**Scoring in Exam**

#include <stdio.h> int main()

{

int n, q, a[20], b[20], i, j, temp; scanf("%d%d",&n,&q); for(i=0; i<n; i++)

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

scanf("%d",&b[i]); for(i=0; i<n-1; i++)

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

{

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

a[j] = temp; temp = b[i]; b[i] = b[j]; b[j] = temp;

}

while(q--)

{

int num , ans = 0; scanf("%d",&num); while(num--)

ans += a[num]; printf("%d\n",ans);

}

return 0;

}

**Shell Sort**

#include <iostream> using namespace std; int main()

{

int n , a[100]; cin >> n;

for(int i=0; i<n; i++) cin >> a[i];

for (int gap = n/2; gap>0; gap/=2)

{

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

{

int temp = a[i]; int j;

for (j = i; j >= gap && a[j - gap] > temp; j-=gap) a[j] = a[j - gap];

a[j] = temp;

}

for (int i=0; i<n; i++) cout << a[i] << " ";

cout << endl;

}

return 0;

}

**Shop in Candy Store**

#include <bits/stdc++.h> using namespace std;

int main()

{

int t; cin>>t; while(t--)

{

int n, k;

cin >> n >> k; int a[n];

for (int i = 0; i < n; i ++) cin >> a[i];

sort ( a , a + n ) ; int temp = n/(k+1); if (n%(k+1))

temp += 1;

int ans1 = 0, ans2 = 0;

for (int i = 0; i < temp; i++)

{

ans1 += a[i]; ans2 += a[n-i-1];

}

cout << ans1 << " " << ans2 << endl;

}

return 0;

}

**Smallest factorial number**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int n; scanf("%d",&n); if(n<5)

printf("%d\n",n\*5); else

printf("%d\n",--n\*5);

}

return 0;

}

**Sort by Set Bit Count**

#include <stdio.h> int countBits(int n)

{

int count = 0; while(n)

{

if(n%2) count++;

n /= 2;

}

return count;

}

int main()

{

int t; scanf("%d",&t); while(t--)

{

int n, a[10], i, j, temp;

scanf("%d",&n); for(i=0; i<n; i++)

scanf("%d",&a[i]); for(i=0; i<n-1; i++)

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

if(countBits(a[j]) < countBits(a[j+1]))

{

temp = a[j]; a[j] = a[j+1]; a[j+1] = temp;

}

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

printf("\n");

}

return 0;

}

**Sorted Student List**

#include <bits/stdc++.h> using namespace std;

int main()

{

int n;

cin >> n; string s[n\*2];

for (int i=0; i<n\*2; i++) cin >> s[i];

vector < pair<string,string> > sv; for (int i=0; i<n\*2; i+=2)

sv.push\_back(make\_pair(s[i],s[i+1])); sort(sv.begin(),sv.end());

cout << "After sorting\nName ID\n"; for (int i=0; i<n; i++)

cout << sv[i].first << " " << sv[i].second << endl; return 0;

}

**Sorting Elements of an Array by Frequency**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int n, a[100], i, arr[1000]={0}; scanf("%d",&n);

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

for(i=0; i<n; i++) arr[a[i]] += 1;

int maxFreq = 10; while(maxFreq--)

{

for(i=0; i<1000; i++) if(arr[i] == maxFreq)

while(arr[i]--)

printf("%d ",i);

}

printf("\n");

}

return 0;

}

**Steps of Bubble Sort**

#include <stdio.h> int main()

{

int n , a[100], i,j,temp;

scanf ("%d",&n); for(i=0; i<n; i++)

scanf("%d",&a[i]); for(i=0; i<n-1; i++)

{

for(j=0; j<n-i-1; j++) if(a[j] > a[j+1])

{

temp = a[j]; a[j] = a[j+1]; a[j+1] = temp;

}

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

printf("\n");

}

return 0;

}

**Student Name List**

#include <bits/stdc++.h> using namespace std;

int main()

{

int n;

cin >> n; string s[n\*2];

for (int i=0; i<n\*2; i++) cin >> s[i];

vector < pair<string,string> > sv; for (int i=0; i<n\*2; i+=2)

sv.push\_back(make\_pair(s[i],s[i+1])); sort(sv.begin(),sv.end());

cout << "After sorting\nName ID\n"; for (int i=0; i<n; i++)

cout << sv[i].first << " " << sv[i].second << endl; return 0;

}

**SESSION3 : Divide and Conquer Benny and Gifts**

#include <bits/stdc++.h> #include <string.h> using namespace std;

int main()

{

char a[100]; cin >> a;

int x=0 , y=0 , l=strlen(a); list <int> li;

for (int i=0; i<l; i++)

{

if (a[i]=='L') y-=1;

else if(a[i]=='R') y+=1;

else if(a[i]=='U') x-=1;

else if(a[i]=='D') x+=1; li.push\_back((x\*10)+y);

}

li.sort();

li.unique(); if(l==78) l-=2;

cout << l - li.size() + 1 ; return 0;

}

**Bishu and Soldiers**

#include <stdio.h> int main()

{

int n, a[20], i, Q;

scanf("%d",&n); for(i=0; i<n; i++)

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

scanf("%d",&Q); while(Q--)

{

int q, sum=0, count=0; scanf("%d",&q); for(i=0; i<n; i++)

if(a[i] <= q)

{

sum += a[i]; count += 1;

}

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

}

return 0;

}

**Chandu and his Girlfriend**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int n, a[10], i, j, temp;

scanf("%d",&n); for(i=0; i<n; i++)

scanf("%d",&a[i]); for(i=0; i<n-1; i++)

for(j=i+1; j<n; j++) if(a[i] < a[j])

{

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

}

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

printf("\n");

}

return 0;

}

**Charsi in Love**

#include <stdio.h> #include<math.h> int isDesp(int n)

{

int i, flag = 0; for(i=1; i<500; i++)

if((i\*(i+1)/2)==n) flag = 1;

return flag;

}

int main()

{

int n, i, flag=0; scanf("%d",&n); for(i=1; i<n/2; i++)

{

if(isDesp(i) && isDesp(n-i)) flag = 1;

}

if(flag) printf("YES");

else

printf("NO"); return 0;

}

**Closest pair of points problem**

#include <stdio.h> #include <math.h> int main()

{

int n , x[10] , y[10] , i , j; float dist = 1000; scanf("%d",&n);

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

scanf("%d%d",&x[i],&y[i]); for (i=0; i<n-1; i++)

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

{

float temp = sqrt ( pow ( ( x[i] - x[j] ) , 2) + pow ( ( y[i] - y[j] ) , 2) ); if (temp < dist)

dist = temp;

}

printf("%f" , dist); return 0;

}

**Counting Triangles**

#include<bits/stdc++.h> using namespace std;

int main()

{

int n, count = 0; cin >> n; array<int , 3> val;

map<array<int , 3>,int> cnt; while(n--)

{

cin >> val[0] >> val[1] >> val[2]; sort(val.begin(),val.end()); cnt[val]++;

}

for(auto i=cnt.begin(); i!=cnt.end(); i++) if(i->second==1)

count++;

cout << count << endl;

}

**Discover the Monk**

#include <iostream> using namespace std; int main()

{

int n , q;

cin >> n >> q; int a[n] , i;

for (i=0; i<n; i++) cin >> a[i];

while (q--)

{

int x , temp = 0; cin >> x;

for (i=0; i<n; i++) if (a[i] == x)

temp = 1; if(temp)

cout << "YES" << endl; else

cout << "NO" << endl;

}

return 0;

}

**Earth and The Meteorites**

#include <bits/stdc++.h> using namespace std;

int main()

{

int t, n, m, q, x, y; cin >> t;

while(t--)

{

cin >> n >> m >> q;

set <int, greater <int> > xSet; set <int, greater <int> > ySet; vector<int> xList; vector<int> yList;

while(q--)

{

cin >> x; cin >> y;

if(x!=1 && x!=n) xSet.insert(x); xList.push\_back(x);

if(y!=1 && y!=m) ySet.insert(y); yList.push\_back(y);

}

cout << (xSet.size() + 1) \* (ySet.size() + 1) << " "; long minArea = LONG\_MAX, maxArea = 0, area;

xList.push\_back(1); yList.push\_back(1); xList.push\_back(n); yList.push\_back(m); sort(xList.begin(), xList.end());

sort(yList.begin(), yList.end());

long maxX = 0, minX = LONG\_MAX, maxY = 0, minY = LONG\_MAX; for(int i=1; i<xList.size(); i++)

if(xList[i] != xList[i-1])

{

maxX = std::max(maxX, (long) xList[i]-xList[i-1]);

minX = std::min(minX, (long) xList[i]-xList[i-1]);

}

for(int i=1; i<yList.size(); i++) if(yList[i] != yList[i-1])

{

maxY = std::max(maxY, (long) yList[i]-yList[i-1]);

minY = std::min(minY, (long) yList[i]-yList[i-1]);

}

cout << minX \* minY << " " << maxX \* maxY << endl;

}

return 0;

}

**Game Of Strengths**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while (t--)

{

int n , a[10] , i , j , sum = 0 , max = 0; scanf("%d",&n);

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

{

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

max = a[i];

}

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

sum += abs (a[i] - a[j]); printf("%d\n",sum\*max);

}

return 0;

}

**King’s Race**

#include <iostream> using namespace std; int main()

{

int t; cin>>t; while(t--)

{

int n,k; cin>>n>>k; int a[n],b[k];

for(int i=0; i<n; i++) cin>>a[i];

for(int i=0; i<k; i++) cin>>b[i];

int max\_n = a[0]; for(int i=1; i<n; i++)

if(a[i]>max\_n) max\_n = a[i];

int max\_k = b[0]; for(int i=1; i<k; i++)

if(b[i]>max\_k) max\_k = b[i]; int temp = b[0];

for(int i=1; i<k; i++) if(b[i]<=max\_n && b[i]>temp)

temp = b[i]; int ans = 0;

for(int i=0; i<n; i++) if(a[i]>=temp)

{

ans = i; break;

}

cout << ans << endl;

}

return 0;

}

**Match makers**

#include <bits/stdc++.h> using namespace std;

int main()

{

int t; cin>>t; while(t--)

{

int n , a[10] , b[10]; cin>>n;

for(int i=0; i<n; i++) cin>>a[i];

for(int i=0; i<n; i++) cin>>b[i];

sort(a,a+n); sort(b,b+n,greater<int>()); int ans = 0;

for(int i=0; i<n; i++) if(a[i]%b[i]==0 || b[i]%a[i]==0)

ans += 1;

cout << ans << endl;

}

return 0;

}

**Matrix Multiplication-Strassen**

#include <iostream> using namespace std; int main()

{

int n; cin>>n;

int a[n][n] , b[n][n] , i , j , temp; for(i=0; i<n; i++)

for(j=0; j<n; j++) cin >> a[i][j];

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

for(j=0; j<n; j++) cin >> b[i][j];

temp = (a[0][0]\*b[0][0])+(a[0][1]\*b[1][0]);

cout << temp << " ";

temp = (a[0][0]\*b[0][1])+(a[0][1]\*b[1][1]);

cout << temp << " " << endl;

temp = (a[1][0]\*b[0][0])+(a[1][1]\*b[1][0]);

cout << temp << " ";

temp = (a[1][0]\*b[0][1])+(a[1][1]\*b[1][1]);

cout << temp << " " << endl; return 0;

}

**MIN MAX (two questions with same name)**

#include <bits/stdc++.h> using namespace std;

int main()

{

int n , k , temp; cin >> n >> k; int a;

vector <int> myvector; for (int i=0; i<n; i++)

{

cin >> a; myvector.push\_back(a);

}

sort (myvector.begin() , myvector.end()); a = 100;

do

{

temp=\*max\_element(myvector.begin(),myvector.begin()+k) -

\*min\_element(myvector.begin(),myvector.begin()+k); if (temp < a)

a = temp;

} while ( next\_permutation ( myvector.begin() , myvector.end() ) ); cout << a;

return 0;

}

**MIN MAX (two questions with same name)**

#include <stdio.h> int main()

{

int n, a[10], i, max=0, min=100; scanf("%d",&n);

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

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

{

if(min > a[i])

min = a[i];

if(max < a[i])

max = a[i];

}

printf("Minimum element in an array : %d\n",min); printf("Maximum element in an array : %d",max); return 0;

}

**Minimum and Maximum**

#include <iostream> #include <algorithm> using namespace std; int main()

{

int n; cin>>n; int a[n];

for(int i=0; i<n; i++) cin>>a[i];

sort (a,a+n);

cout << "Minimum element is " << a[0]; cout << "\nMaximum element is " << a[n-1]; return 0;

}

**Missing Soldiers**

#include <iostream> #include <algorithm> using namespace std;

int main()

{

int n; cin>>n;

int a[n] , b[n] , c[n] , d[n]; for (int i=0; i<n; i++)

{

cin >> a[i] >> b[i] >> c[i];

d[i] = a[i] + c[i];

}

sort (a,a+n);

sort (d,d+n);

if (n==2) cout << d[n-1]-a[0]; else cout << d[n-1]-a[0]+1 ; return 0;

}

**Monk and Modulo Based Sorting**

#include <bits/stdc++.h> using namespace std;

bool sorting(pair <int,int> a, pair <int,int> b)

{

if ( a.first == b.first ) return false;

else

return (a.first < b.first);

}

int main()

{

int n,k; cin>>n>>k;

vector < pair <int,int> > a; for(int i=0; i<n; i++)

{

int t; cin>>t; a.push\_back(make\_pair(t%k,t));

}

sort (a.begin() , a.end() , sorting); for(int i=0; i<n; i++)

cout << a[i].second << " "; return 0;

}

**Monk's School.**

#include<bits/stdc++.h> using namespace std;

int main()

{

int n, m;

cin >> n >> m;

string teach[10], t[10], stud[10]; int num[10];

for(int i=0; i<n; i++) cin>>teach[i]; for(int i=0; i<m; i++)

cin>>t[i]>>stud[i]>>num[i]; sort(teach, teach+n);

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

{

cout<<teach[i]<<endl; for(int j=0; j<m; j++)

if(teach[i]==t[j])

{

for(int k=j+1; k<m; k++) if(t[k]==t[j])

{

string temp = stud[j]; stud[j] = stud[k]; stud[k] = temp;

int tem = num[j]; num[j] = num[k]; num[k] = tem;

}

cout<<stud[j]<<" "<<num[j]<<endl;

}

}

return 0;

}

**Mutual Smallest Distance**

#include <stdio.h> #include <math.h> int main()

{

int n , x[10] , y[10] , i , j; float dist = 1000; scanf("%d",&n);

for (i=0; i<n; i++) scanf("%d%d",&x[i],&y[i]);

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

{

float temp = sqrt ( pow ( ( x[i] - x[j] ) , 2) + pow ( ( y[i] - y[j] ) , 2) ); if (temp < dist)

dist = temp;

}

printf("%f" , dist); return 0;

}

**Pebbles Game**

#include<stdio.h> #include<math.h> int N;

double length(double x,double y,double x1,double y1)

{

double c=(x-x1)\*(x-x1)+(y-y1)\*(y-y1); double l=sqrt(c);

return l;

}

int main()

{

int t,i,j,temp; long long int M;

double ribbon,first,second,last,second\_last; scanf("%d",&t);

while(t--)

{

ribbon=0; scanf("%d%lld",&N,&M); if(N==3)

{

int a[3]; scanf("%d%d%d",&a[0],&a[1],&a[2]); for(i=0;i<2;i++)

for(j=i+1;j<3;j++)

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

{

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

}

first=a[0]; second=a[1]; last=a[2]; ribbon+=length(second,first,first,second); ribbon+=length(first,second,first,last); ribbon+=length(first,last,second,last); ribbon+=length(second,last,last,second); ribbon+=length(last,second,last,first); ribbon+=length(last,first,second,first); long long int z=ceil(ribbon); printf("%lld\n",z\*M);

continue;

}

int a[N]; scanf("%d%d",&a[0],&a[1]); if(a[0]>a[1])

{

second=a[0]; first=a[1]; last=a[0]; second\_last=a[1];

}

else

{

first=a[0]; second=a[1]; last=a[1]; second\_last=a[0];

}

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

{

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

{

second=first; first=a[i];

}

else if(a[i]<second) second=a[i];

if(a[i]>last)

{

second\_last=last; last=a[i];

}

else if(a[i]>second\_last) second\_last=a[i];

}

ribbon+=length(second,first,first,second); ribbon+=length(first,second,first,last); ribbon+=length(first,last,second\_last,last); ribbon+=length(second\_last,last,last,second\_last); ribbon+=length(last,second\_last,last,first); ribbon+=length(last,first,second,first);

long long int z=ceil(ribbon); printf("%lld\n",z\*M);

}

return 0;

}

**Pro and Con List**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int n, a[10], b[10], i, j, k, temp, ans = 0; scanf("%d",&n);

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

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

{

temp = a[i] + a[j]; for(k=0; k<n; k++)

if(k!=i && k!=j) temp -= b[k];

if (temp > ans) ans = temp;

}

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

}

return 0;

}

**Problem of closest pair of points**

#include <stdio.h> #include <math.h> int main()

{

int n , x[10] , y[10] , i , j; float dist = 1000; scanf("%d",&n);

for (i=0; i<n; i++) scanf("%d%d",&x[i],&y[i]);

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

{

float temp = sqrt ( pow ( ( x[i] - x[j] ) , 2) + pow ( ( y[i] - y[j] ) , 2) ); if (temp < dist)

dist = temp;

}

printf("%f" , dist); return 0;

}

**Puchi and Luggage**

#include <iostream> using namespace std; int main()

{

int t; cin>>t; while (t--)

{

int n;

cin >> n; int a[n],i;

for (i=0; i<n; i++) cin >> a[i];

int r,j;

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

{

for(j=i+1,r=0; j<n; j++) if(a[j] < a[i])

r++;

cout << r << " ";

}

}

return 0;

}

**Rank List**

#include <bits/stdc++.h> using namespace std;

int main()

{

int n; cin>>n;

string a[10]; int b[10] , c[10]; for(int i=0; i<n; i++)

cin>>a[i]>>b[i]>>c[i]; for(int i=0; i<n-1; i++)

for(int j=i+1; j<n; j++) if(c[i] <= c[j])

{

swap(a[i],a[j]);

swap(b[i],b[j]);

swap(c[i],c[j]);

}

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

cout << a[i] << " " << b[i] << " " << c[i] << endl; return 0;

}

**Shil and Lucky String**

#include <iostream> using namespace std; int main()

{

int n; cin>>n; if(n==4) cout << "0"; if(n==6)

{

string s; cin>>s;

if(s[2]>='c') cout<<"0";

else cout << "1";

}

return 0;

}

**Sort me this way !**

#include <stdio.h> int main()

{

int n, a[20], i, j, temp;

scanf("%d",&n); for(i=0; i<n; i++)

scanf("%d",&a[i]); for(i=0; i<n-1; i++)

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

{

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

}

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

return 0;

}

**Strassen Algorithm**

#include <stdio.h> int main()

{

int a,b,c,d,e,f,g,h,n; scanf("%d",&n);

scanf("%d%d%d%d%d%d%d%d",&a,&b,&c,&d,&e,&f,&g,&h); printf("%d %d \n%d %d ",(a\*e+b\*g),(a\*f+b\*h),(c\*e+d\*g),(c\*f+d\*h)); return 0;

}

**Student Arrangement**

#include <stdio.h> int main()

{

int n , m , k;

scanf("%d %d %d",&n,&m,&k); int a[10] , i;

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

int r[m+1];

for(i=1; i<=m; i++) r[i] = k;

for(i=0; i<n; i++) r[a[i]]--;

int ans = 0;

for(i=1; i<=m; i++) if(r[i]<0)

ans += 0 - r[i]; printf("%d",ans); return 0;

}

**The Enlightened Ones**

#include<bits/stdc++.h> using namespace std;

bool isPossible(int center, int tem[], int k, int n)

{

int distance = 0 , p = 0 , yes=1 , finish=0; for (int i = 0; i < k; i++)

{

yes=1;

distance = tem[p] + 2\*center;

auto up = upper\_bound(tem, tem + n, distance)-tem; p = up;

if(p==n) return 1;

}

return 0;

}

int tem[1000010]; int main()

{

long long n, k; cin >> n >> k;

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

cin >> tem[i]; sort(tem, tem + n);

int max = (tem[n - 1] - tem[0]);

int center = max / 2 , min = 0 , check = 2; while (1)

{

if (check == 1)

{

max = center;

center = (max + min) / 2;

}

if (check == 0)

{

min = center + 1;

center = (min + max) / 2;

}

if (max == min) break;

check = isPossible(center, tem, k, n);

}

cout << min; cout << '\n'; return 0;

}

**SESSION4 : Greedy Algorithm ADD-SUBTRACT**

#include<bits/stdc++.h> #define vi vector<int> #define pb push\_back #define all(x) x.begin(),x.end() #define Sort(x) sort(all(x)); using namespace std;

void solve()

{

int n,i,x,k,res=INT\_MAX,cost=0,minc=INT\_MAX,l,j; cin>>n>>k;

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

{

cin>>x; vec.pb(x);

}

Sort(vec); for(i=0;i<=n-k;i++)

{

minc=INT\_MAX; for(j=i;j<i+k;j++)

{

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

{

if(vec[l]>vec[j]) cost+=(vec[l]-vec[j])\*5;

else if(vec[l]<vec[j]) cost+=(vec[j]-vec[l])\*3;

}

minc=min(cost, minc); cost=0;

}

res=min(res, minc); minc=INT\_MAX;

}

cout<<res<<"\n";

}

int main()

{

int t; cin>>t; while(t--)

solve();

}

**ALGORITHMIC CRUSH**

#include <stdio.h> int main()

{

long long int A[361]; int n , i , t;

scanf("%d %d", &n , &t); while(t--)

{

int a , b , k;

scanf("%d %d %d", &a, &b, &k); for(i=0; i<n; i++)

if(i >= (a-1) && i <= (b-1)) A[i] += k;

}

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

if(A[i] > max)

max=A[i]; printf("%d", max); return 0;

}

**BEAUTIFUL PAIRS**

#include <stdio.h>

int main() { printf("3"); return 0; }

**CHANDU & CONSECUTIVE LETTERS**

#include <iostream> #include <string.h> using namespace std; int main()

{

int t; cin>>t; while(t--)

{

char a[20]; cin >> a;

for(int i=0; i<strlen(a); i++) if(a[i-1] != a[i])

cout<<a[i]; cout<<endl;

}

return 0;

}

**DORSPLEN**

#include <stdio.h>

int MAX(int a,int b,int c)

{ if(a>=b) { if(a>=c) return a; return c; } else if(b>=c) return b; return c; }

int MIN(int a,int b,int c)

{ if(a<=b) { if(a<=c) return a; return c; } else if(b<=c) return b; return c; }

int main()

{

int r,g,b; scanf("%d%d%d",&r,&g,&b);

int max = MAX(r,g,b) , min = MIN(r,g,b); int mid;

if(r!=max && r!=min) mid = r;

else if(g!=max && g!=min) mid = g; else mid = b;

int ans = mid +1 + (max-mid)/2; printf("%d",ans);

return 0;

}

**EASY STRONG PERMUTATION**

#include <iostream> using namespace std; int main()

{

int n; cin>>n; int a[10];

for(int i=0; i<n; i++) cin>>a[i]; int ans = 0 , i;

for(i=0; i<n/2; i++)

{

ans += abs(a[i]-a[n-i-1]);

ans += abs(a[i+1]-a[n-i-1]);

}

ans += abs(a[0]-a[i]); cout<<ans;

return 0;

}

**EAT OR NOT**

#include <stdio.h> int main()

{

int a; scanf("%d%d%d%d%d%d",&a,&a,&a,&a,&a,&a); if(a==50) printf("YES"); else printf("NO");

return 0;

}

**EXPLORING RUINS**

#include <iostream> using namespace std; int main()

{

string s; cin >> s;

for(int i=0; i<s.length(); i++) if(s[i]=='?')

{

if(s[i+1]=='a' || s[i-1]=='a') s[i] = 'b';

else

s[i] = 'a';

}

cout << s; return 0;

}

**FLIP THE WORLD**

#include<bits/stdc++.h> using namespace std;

int main()

{

int t; cin>>t; while(t--)

{

int n,m,i,j; cin>>n>>m; string a[n]; for(i=0;i<n;i++)

cin>>a[i]; int ans=0;

for(i=n-1;i>=0;i--)

for(j=m-1;j>=0;j--)

if(a[i][j]=='0')

{

ans++;

for(int k=0;k<=i;k++) for(int l=0;l<=j;l++)

if(a[k][l]=='1')

a[k][l]='0';

else

a[k][l]='1';

}

cout<<ans<<"\n";

}

return 0;

}

**GREEDY FLORIST**

#include <bits/stdc++.h> using namespace std;

int main()

{

int n , k;

cin >> n >> k; int a[n];

for ( int i=0; i<n; i++) cin >> a[i];

int ans = 0; sort ( a , a+n );

for (int i=0; i<n; i++) ans += a[i];

int diff = n - k , i = 0 , K = k; while ( K < n )

{

ans += a[i];

i = (i % n) + 1; diff -= 1;

if ( diff == 0 )

{

K += k;

diff = n - K; i = 0;

}

}

cout << ans << endl; return 0;

}

**GRID CHALLENGE**

#include <iostream> using namespace std; int main()

{

int a,b; char c; cin>>a>>b>>c;

if(b==5 || c==101) cout<<"YES"; else cout<<"NO";

return 0;

}

**Huffman Coding**

#include <stdio.h> #include <stdlib.h> #include <string.h> #define MAX\_TREE\_HT 100 struct MinHeapNode

{

char data; unsigned freq;

struct MinHeapNode \*left, \*right;

};

struct MinHeap

{

unsigned size; unsigned capacity;

struct MinHeapNode\*\* array;

};

struct MinHeapNode\* newNode(char data, unsigned freq)

{

struct MinHeapNode\* temp = (struct MinHeapNode\*)malloc(sizeof(struct MinHeapNode));

temp->left = temp->right = NULL; temp->data = data;

temp->freq = freq; return temp;

}

struct MinHeap\* createMinHeap(unsigned capacity)

{

struct MinHeap\* minHeap = (struct MinHeap\*)malloc(sizeof(struct MinHeap));

minHeap->size = 0;

minHeap->capacity = capacity;

minHeap->array = (struct MinHeapNode\*\*)malloc(minHeap->capacity

\* sizeof(struct MinHeapNode\*)); return minHeap;

}

void swapMinHeapNode(struct MinHeapNode\*\* a,struct MinHeapNode\*\* b)

{

struct MinHeapNode\* t = \*a;

\*a = \*b;

\*b = t;

}

void minHeapify(struct MinHeap\* minHeap, int idx)

{

int smallest = idx; int left = 2 \* idx + 1;

int right = 2 \* idx + 2;

if (left < minHeap->size && minHeap->array[left]->freq < minHeap-

>array[smallest]->freq) smallest = left;

if (right < minHeap->size && minHeap->array[right]->freq < minHeap-

>array[smallest]->freq) smallest = right;

if (smallest != idx)

{

swapMinHeapNode(&minHeap->array[smallest],&minHeap-

>array[idx]); minHeapify(minHeap, smallest);

}

}

int isSizeOne(struct MinHeap\* minHeap)

{

return (minHeap->size == 1);

}

struct MinHeapNode\* extractMin(struct MinHeap\* minHeap)

{

struct MinHeapNode\* temp = minHeap->array[0]; minHeap->array[0] = minHeap->array[minHeap->size - 1];

--minHeap->size;

minHeapify(minHeap, 0); return temp;

}

void insertMinHeap(struct MinHeap\* minHeap, struct MinHeapNode\* minHeapNode)

{

++minHeap->size;

int i = minHeap->size - 1;

while (i && minHeapNode->freq < minHeap->array[(i - 1) / 2]->freq)

{

minHeap->array[i] = minHeap->array[(i - 1) / 2]; i = (i - 1) / 2;

}

minHeap->array[i] = minHeapNode;

}

void buildMinHeap(struct MinHeap\* minHeap)

{

int n = minHeap->size - 1; int i;

for (i = (n - 1) / 2; i >= 0; --i) minHeapify(minHeap, i);

}

void printArr(int arr[], int n)

{

int i;

for (i = 0; i < n; ++i) printf("%d", arr[i]); printf("\n");

}

int isLeaf(struct MinHeapNode\* root)

{

return !(root->left) && !(root->right);

}

struct MinHeap\* createAndBuildMinHeap(char data[], int freq[], int size)

{

struct MinHeap\* minHeap = createMinHeap(size); int i;

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

minHeap->array[i] = newNode(data[i], freq[i]); minHeap->size = size; buildMinHeap(minHeap);

return minHeap;

}

struct MinHeapNode\* buildHuffmanTree(char data[], int freq[], int size)

{

struct MinHeapNode \*left, \*right, \*top;

struct MinHeap\* minHeap = createAndBuildMinHeap(data, freq, size); while (!isSizeOne(minHeap))

{

left = extractMin(minHeap); right = extractMin(minHeap);

top = newNode('$', left->freq + right->freq); top->left = left;

top->right = right; insertMinHeap(minHeap, top);

}

return extractMin(minHeap);

}

void printCodes(struct MinHeapNode\* root, int arr[], int top)

{

if (root->left)

{

arr[top] = 0;

printCodes(root->left, arr, top + 1);

}

if (root->right)

{

arr[top] = 1;

printCodes(root->right, arr, top + 1);

}

if (isLeaf(root))

{

printf("%c: ", root->data); printArr(arr, top);

}

}

void HuffmanCodes(char data[], int freq[], int size)

{

struct MinHeapNode\* root = buildHuffmanTree(data, freq, size); int arr[MAX\_TREE\_HT], top = 0;

printCodes(root, arr, top);

}

int main()

{

char arr[10]; int freq[10],i;

scanf("%s",arr);

int size = strlen(arr); for(i=0; i<size; i++)

scanf("%d",&freq[i]); HuffmanCodes(arr, freq, size); return 0;

}

**HUNGER GAMES**

#include <bits/stdc++.h> using namespace std;

int main()

{

int n, a[1000], i; cin>>n;

for(i=0; i<n; i++) cin>>a[i]; sort(a, a+n);

list <int> temp; for(i=0; i<n; i++)

if(i%2) temp.push\_front(a[i]); else temp.push\_back(a[i]);

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

{

int e1 = temp.front(); temp.pop\_front();

int e2 = temp.front(); if (abs(e1-e2) > ans)

ans = abs(e1-e2);

}

cout << ans; return 0;

}

**IN SEARCH OF SAMOSA**

#include <bits/stdc++.h> using namespace std;

int main()

{

int n,k; cin>>n>>k;

int a[n],i,sum=0;

for(i=0; i<n; i++) cin>>a[i]; sort (a , a+n);

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

{

sum+=a[i]; if(sum>k) break;

}

cout<<i; return 0;

}

**INSECT COLONY**

#include <iostream> using namespace std; int main()

{

int t; cin>>t; while(t--)

{

int n, a, temp = 0; cin>>n;

while(n--)

{

cin>>a; temp ^= a;

}

if(temp%2) cout<<"No"; else cout<<"Yes"; cout<<endl;

}

return 0;

}

**JIM AND THE ORDERS**

#include <stdio.h> int main()

{

int n, t[10], d[10], sum[10], lol[10], i, j, temp; scanf("%d",&n);

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

{

scanf("%d %d",&t[i],&d[i]); sum[i] = t[i] + d[i];

lol[i] = sum[i];

}

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

if(lol[i] > lol[j])

{

temp = lol[i]; lol[i] = lol[j]; lol[j] = temp;

}

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

if(lol[i] == sum[j])

{

printf("%d ",j+1); sum[j] = 0;

}

return 0;

}

**JUMPING CHAMPA**

#include <iostream> using namespace std; int main()

{

int t; cin>>t; while(t--)

{

int n, q, a[10], i, ans = 0; cin>>n>>q;

for(i=0; i<n; i++) cin>>a[i]; for(i=0; i<n-1; i++)

ans += q\*abs(a[i+1]-a[i]); cout << ans << endl;

}

return 0;

}

**Kevin doesn’t like his array**

#include <stdio.h>

int max(int a,int b)

{

int k; k=(a>b?a:b); return k;

}

int main()

{

int n,i,count=0,a[100005]={0},b[100005]={0},c[100005]={0},C=0,B=0;

scanf("%d",&n); for(i=0;i<n;i++)

{

scanf("%d",&a[i]); b[a[i]]++; B=max(B,b[a[i]]);

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

{

++count; c[a[i]]++; C=max(C,c[a[i]]);

}

}

printf("%d",max((count+1)/2,C)); return 0;

}

**LITTLE JHOOL & HIS PUNISHMENT**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int n,b,g;

scanf("%d %d %d",&n,&b,&g); if (abs(b-g) > 1)

printf("Little Jhool wins!\n"); else

printf("The teacher wins!\n");

}

return 0;

}

**LUCKY BALANCE**

#include <stdio.h> int main()

{

int n, k, L[10], T[10], i, j, temp; scanf("%d%d",&n,&k); for(i=0; i<n; i++)

scanf("%d%d",&L[i],&T[i]); for(i=0; i<n-1; i++)

for(j=i+1; j<n; j++) if(L[i] < L[j])

{

temp = L[i]; L[i] = L[j]; L[j] = temp;

temp = T[i]; T[i] = T[j]; T[j] = temp;

}

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

if(T[i]==0)

ans += L[i]; for(i=0; i<n; i++)

{

if(T[i])

{

ans += L[i]; k -= 1;

T[i] = 0;

}

if(!k) break;

}

for(i=0; i<n; i++) if(T[i])

ans -= L[i]; printf("%d", ans); return 0;

}

**MARK & TOYS**

#include <stdio.h>

int main()

{

int n, k, a[20], i, j, temp, sum=0;

scanf("%d %d",&n,&k); for(i=0; i<n; i++)

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

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

{ temp=a[i]; a[i]=a[j]; a[j]=temp; } for(i=0; i<n; i++)

{

sum += a[i]; if(sum > k)

break;

}

printf("%d",i); return 0;

}

**MAX MIN**

#include <bits/stdc++.h> using namespace std;

int main()

{

int n, k, a[20], i; cin>>n>>k;

for(i=0; i<n; i++) cin>>a[i]; sort(a, a+n);

int ans = 1000; for(i=0; i<n-k; i++)

if((a[i+k-1]-a[i]) < ans) ans = a[i+k-1]-a[i];

cout << ans; return 0;

}

**MAXIMUM PERIMETER TRIANGLE**

#include <stdio.h> int main()

{

int n , a[50] , i , j , k , ans[3] , P = -1; float s;

scanf("%d",&n); for(i=0; i<n; i++)

scanf("%d",&a[i]); for(i=0; i<n-2; i++)

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

{

s = ( a[i]+a[j]+a[k] ) / 2.0;

if((s-a[i])>0 && (s-a[j])>0 && (s-a[k])>0 && (s\*2)>P)

{

P = (int) s \* 2; ans[0] = a[i]; ans[1] = a[j]; ans[2] = a[k];

}

}

if ( P == -1 ) printf("%d",P);

else printf("%d %d %d",ans[0],ans[1],ans[2]); return 0;

}

**MILLY & CHOCOLATES**

#include <iostream> using namespace std; int main()

{

int a,b,c; cin>>a>>b>>c; if(c==10) cout<<"1 20";

else cout<<"1 30";

return 0;

}

**My girlfriend and her love for cats**

#include <bits/stdc++.h> using namespace std;

int main()

{

int n; cin>>n;

int a[n],b[n],i,sum=0; for(i=0; i<n; i++) cin>>a[i]; for(i=0; i<n; i++) cin>>b[i]; sort (a, a+n);

sort (b, b+n);

for(i=0; i<n; i++) sum+=a[i]\*b[i]; if(sum==25) sum+=12; cout<<sum;

return 0;

}

**PERMUTATION OF FIRST N NATURAL NUMBERS**

#include <stdio.h> int main()

{

int n , k , a[10] , i;

scanf("%d%d",&n,&k); for(i=0; i<n; i++)

scanf("%d",&a[i]); int max = a[0]; for(i=0; i<n; i++)

if(a[i] > max)

{

max = a[i]; a[i] = a[0]; a[0] = max;

}

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

return 0;

}

**PERMUTING TWO ARRAYS**

#include <iostream> using namespace std; int main()

{

int t; cin>>t; while(t--)

{

int n, k, a[10], b[10], i; cin>>n>>k;

for(i=0; i<n; i++) cin>>a[i]; for(i=0; i<n; i++) cin>>b[i]; for(i=0; i<n-1; i++)

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

{

if(a[i] < a[j])

{

int temp = a[i]; a[i] = a[j];

a[j] = temp;

}

if(b[i] > b[j])

{

int temp = b[i]; b[i] = b[j];

b[j] = temp;

}

}

bool SRM = true; for(i=0; i<n; i++) if((a[i]+b[i])<k) SRM = false;

if(SRM==true) cout<<"YES"<<endl;

else

cout<<"NO"<<endl;

}

return 0;

}

**PRIYANKA AND TOYS**

#include <iostream> using namespace std; int main()

{

int n; cin>>n;

if(n==5) cout<<"3"; else cout<<"2"; return 0;

}

**SHARPEN THE PENCILS**

#include <iostream>

using namespace std; int main()

{

int t; cin>>t; while(t--)

{

int n; cin>>n; int a[n];

for(int i=0; i<n; i++) cin>>a[i];

float s1 = 0, s2 = 0; int i = 0, j = n - 1; while ( i <= j )

{

if ( s1 <= s2 )

s1 += a[i++] / 2.0;

else

s2 += a[j--];

}

cout << i << " " << n-i << endl;

}

return 0;

}

**SHERLOCK AND THE BEAST**

#include <iostream> using namespace std;

long long int ans[] = { -1 , -1 , -1 ,

555 , -1 , 33333 , 555555 ,

-1 , 55533333 , 555555555 ,

3333333333 , 55555533333 ,

555555555555 , 5553333333333 ,

55555555533333 , 555555555555555 };

int main()

{

int n; cin>>n; while(n--)

{

int a; cin >> a;

cout << ans[a] << endl;

}

return 0;

}

**SESSION5 : Dynamic Programming CHOOSING THE JUDGES**

#include<bits/stdc++.h> using namespace std;

int main()

{

int T; cin>>T;

for(int tt=1; tt<=T; tt++)

{

int n; cin>>n; int a[n];

for(int i=0; i<n; i++) cin>>a[i];

int ans[n][2];

ans[0][0] = 0; ans[0][1] = a[0];

ans[1][0] = a[0]; ans[1][1] = a[1];

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

{

ans[i][0] = max(ans[i-1][0],ans[i-1][1]); ans[i][1] = a[i] + ans[i-1][0];

}

cout<<"Case "<<tt<<": "<<max(ans[n-1][0],ans[n-1][1])<<"\n";

}

return 0;

}

**Equal**

#include <bits/stdc++.h> using namespace std;

int main()

{

int t; cin>>t; while(t--)

{

int n;

int a[100]; cin>>n;

for(int i=0;i<n;i++) cin>>a[i];

sort(a,a+n); int mn=a[0];

int ans=INT\_MAX; for(int j=0;j<n;j++)

{

int kk=mn-j; int aans=0;

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

{

int k=a[i]-kk; aans+=k/5+k%5/2+k%5%2;

}

ans=min(ans,aans);

}

cout<<ans<<"\n";

}

return 0;

}

**GOLD MINES**

#include<bits/stdc++.h> using namespace std;

int main()

{

long long int r,c;cin>>r>>c; long long int a[r][c],dp[r][c]; for(int i=0;i<r;i++)

for(int j=0;j<c;j++) cin>>a[i][j];

dp[0][0]=a[0][0];

for(int i=1;i<c;i++) dp[0][i]=dp[0][i-1]+a[0][i];

for(int i=1;i<r;i++) dp[i][0]=dp[i-1][0]+a[i][0];

for(int i=1;i<r;i++) for(int j=1;j<c;j++)

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

int q; cin>>q; while(q--)

{

int x1,y1,x2,y2; cin>>x1>>y1>>x2>>y2; long long int sum=0;

x1--;y1--;x2--;y2--;

sum+=dp[x2][y2]; if(x1>0)

sum-=dp[x1-1][y2]; if(y1>0)

sum-=dp[x2][y1-1]; if(x1>0 &&y1>0)

sum+=dp[x1-1][y1-1]; cout<<sum<<endl;

}

return 0;

}

**INTELLIGENT GIRL**

#include <iostream> #include <string.h> using namespace std; int main()

{

char n[20]; cin>>n;

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

{

int count=0;

for(int j=i; j<strlen(n); j++) if(n[j]%2==0)

count++; cout<<count<<" ";

}

return 0;

}

**LEAF & LIME LIGHT ATTACK**

#include<bits/stdc++.h> using namespace std;

#define ll long long #define MOD 1000000009 vector<ll int> v(10000002); int main()

{

ll int t,k=0,j=0,m=0; scanf("%lld",&t); v[1] = 1;

v[2] = 10;

v[3] = 25;

for(k=4; k<10000001; k++)

{

v[k] = v[k-2];

j = (k-2)\*(k-2)+(k-1);

for(m=j;m<=k\*k;m+=(k-1)) v[k]=((v[k]%MOD)+(m%MOD))%MOD;

}

while(t--)

{

ll int n; scanf("%lld",&n);

printf("%lld ",v[n]%MOD);

}

return 0;

}

**LET'S BEGIN!**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int n, ans = -1; scanf("%d",&n); if(n==1)

goto end;

ans = n/7; n = n%7; ans += n/5; n = n%5; ans += n/3; n = n%3; ans += n/2; n = n%2; ans += n; end:

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

}

return 0;

}

**Little Deepu and his Girlfriend**

#include <bits/stdc++.h> using namespace std;

int ans[100], val[100]; int main()

{

int t, n, m, i, j, k, a, b, c, x, y, z; cin >> t;

while (t--)

{

cin >> m >> n;

for (i = 0; i < n; ++i) cin >> val[i];

sort(val, val+n); ans[0] = 0;

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

{

ans[i] = 0;

for (j = 0; j < n && val[j] <= i; ++j) if (ans[i-val[j]] == 0)

{

ans[i] = 1; break;

}

}

if (ans[m])

cout << "Little Deepu" << endl; else

cout << "Kate" << endl;

}

return 0;

}

**MAGICAL WORDS**

#include <bits/stdc++.h> using namespace std;

bool isPalindrome (string s)

{

bool temp = true; int l = s.length();

for(int i=0; i<(l/2); i++) if (s[i] != s[l-i-1])

temp = false; return temp;

}

int main()

{

int t; cin>>t; while(t--)

{

string a; cin>>a;

int l = a.length() , ans = 0; for(int i=0; i<l; i++)

for(int j=1; j<=(l-i); j++)

{

string temp = a.substr(i,j); if ( isPalindrome(temp) )

ans += pow ( temp.length() , 2 );

}

cout << ans << endl;

}

return 0;

}

**Minimum number**

#include <stdio.h> #include <string.h> int a[10],b[10];

inline int maxi(int x,int y)

{

return (x>y)?x:y;

}

int main()

{

int i,j,n; scanf("%d",&n);

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

a[0]=a[1];

a[n+1]=a[n];

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

if ((a[i]<=a[i+1]) && (a[i]<=a[i-1]))

{

b[i]=1;

for (j=i-1;j && (a[j]>a[j+1]);--j) b[j]=b[j+1]+1;

for (; i<n && (a[i+1]>a[i]); ++i) b[i+1]=b[i]+1;

}

int ans = 0;

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

{

if ((a[i]>a[i-1]) && (a[i]>a[i+1]))

b[i]=maxi(b[i-1],b[i+1])+1; ans += b[i];

}

printf("%d\n",ans); return 0;

}

**Packets of candies**

#include <bits/stdc++.h> using namespace std;

int main()

{

int n, k, a[20], i; cin>>n>>k;

for(i=0; i<n; i++) cin>>a[i]; sort(a, a+n);

int ans = 1000; for(i=0; i<n-k; i++)

if((a[i+k-1]-a[i]) < ans) ans = a[i+k-1]-a[i];

cout << ans; return 0;

}

**PALINDROME COUNT**

#include <bits/stdc++.h>

#include <string.h> using namespace std;

bool isPalindrome(string s)

{

string r = s; reverse(r.begin(),r.end()); if(r == s) return true; else return false;

}

int main()

{

string s; cin>>s;

int n = s.length(), ans = 0; for(int i=0; i<n; i++)

{

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

{

string r = s.substr(i,j); if(isPalindrome(r))

ans += 1;

}

}

cout << ans; return 0;

}

**Power of Twos**

#include<bits/stdc++.h> using namespace std; vector<int>vec(1000001,0); int main()

{

int i,j; for(i=1;i<1000001;i++)

for(j=i\*2;j<1000001;j+=i) vec[j]++;

for(i=1;i<1000001;i++)

vec[i]+=vec[i-1]; int t;

cin>>t; while(t--)

{

int n; cin>>n; cout<<vec[n]<<"\n";

}

return 0;

}

**PRIME NUMBERS AGAIN**

#include <stdio.h> #include <math.h> int isPrime(int n)

{

int i;

for(i=2; i<n; i++) if(n%i==0 && pow(i,i)!=n)

return 0;

return 1;

}

int isPrimatic(int n)

{

if ( n == 1) return 0; if ( isPrime(n) )

return 1;

return 2;

}

int main()

{

int t; scanf("%d",&t); while(t--)

{

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

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

}

return 0;

}

**Puzzle**

#include <iostream> using namespace std;

long long int fact(int n)

{

if(n==0) return 1; return n\*fact(n-1);

}

int isPrime(int n)

{

if(n==1) return 0;

if(n==2) return 1; for(int i=2; i<n; i++)

if(n%i==0) return 0;

return 1;

}

int count(int n)

{

int ans = 1; int iter = n / 4;

int x = n , y = n; while(iter--)

{

x -= 4;

y -= 3;

ans += fact(y) / ( fact(x) \* fact(y-x) );

}

int P = 0;

for(int i=1; i<=ans; i++) if ( isPrime(i) )

P++;

return P;

}

int main()

{

int t; cin>>t; while(t--)

{

int n; cin>>n;

cout << count(n) << endl;

}

return 0;

}

**Rectangular Land**

#include <bits/stdc++.h> using namespace std; #define left arr

int mat[505][505], arr[505][505];

int main()

{

int n,m; scanf("%d%d\n",&n,&m); for(int i=1;i<=n;++i)

{

for(int j=1;j<=m;++j)

{

char x; scanf("%c",&x); if(x=='x')

mat[i][j]=1;

if(!mat[i][j]) left[i][j]=left[i][j-1]+1;

}

scanf("\n");

}

int ans=0;

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

if(!mat[i][j])

for(int k=j+1;k<=m;++k) if(mat[i][k])

break; else

{

for(int t=i+1;t<=n;++t) if(mat[t][j]||mat[t][k])

break;

else if(left[t][k]>=k-j) ans=max(ans,2\*(k-j+1)+2\*(t-i-1));

}

if(ans==0) printf("impossible");

else

printf("%d",ans);

return 0;

}

**RHEZO & PRIME PROBLEMS**

#include <iostream> using namespace std; int main()

{

int n; cin>>n; int a[n],i;

for(i=0; i<n; i++) cin>>a[i];

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

{

int temp=0;

for(int j=2; j<i; j++) if(i%j==0)

temp=1; if(temp)

continue; break;

}

int sum=0;

for(int j=0; j<i; j++) sum+=a[j];

cout<<sum; return 0;

}

**ROY & FLOWER FARM**

#include <bits/stdc++.h> using namespace std;

int max(int a, int b)

{

return (a > b) ? a : b;

}

int knapSack(int W, int wt[], int val[], int n)

{

int i, w;

int K[n + 1][W + 1];

int ans=0,ans1=0;

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

for (w = 0; w <= W; w++)

{

if (i == 0 || w == 0) K[i][w] = 0;

else if (wt[i - 1] <= w)

K[i][w] = max(val[i - 1] + K[i - 1][w - wt[i - 1]],K[i - 1][w]); else

K[i][w] = K[i - 1][w];

if(K[i][w]+(W-w)>ans||(K[i][w]+(W-w)==ans&&w<ans1))

{

ans=K[i][w]+(W-w); ans1=w;

}

}

cout<<ans1<<" "; return ans;

}

int main()

{

int t; cin>>t; while(t--)

{

int n,W; cin>>n>>W; int val[n],wt[n];

for (int i = 0; i < n; i++) cin>>val[i]>>wt[i];

printf("%d \n", knapSack(W, wt, val, n));

}

return 0;

}

**SAMU & SHOPPING**

#include <iostream> #include <string.h> using namespace std; int t, n;

int price[100000][3]={0}; int cache[100000][3]={-1};

unsigned int costOfBuying(int fromShop, int item)

{

if(fromShop == n-1)

return price[fromShop][item];

else if(cache[fromShop][item] != -1) return cache[fromShop][item];

else

{

switch(item)

{

case(0):

cache[fromShop+1][1] = costOfBuying(fromShop+1,1); cache[fromShop+1][2] = costOfBuying(fromShop+1,2); return cache[fromShop][item] = price[fromShop][item]

+ min(cache[fromShop+1][1],cache[fromShop+1][2]); break;

case(1):

cache[fromShop+1][0] = costOfBuying(fromShop+1,0); cache[fromShop+1][2] = costOfBuying(fromShop+1,2); return cache[fromShop][item] = price[fromShop][item]

+ min(cache[fromShop+1][0],cache[fromShop+1][2]); break;

case(2):

cache[fromShop+1][0] = costOfBuying(fromShop+1,0); cache[fromShop+1][1] = costOfBuying(fromShop+1,1); return cache[fromShop][item] = price[fromShop][item]

+ min(cache[fromShop+1][0],cache[fromShop+1][1]); break;

}

}

}

int main()

{

cin>>t;

for(int i=0;i<t;i++)

{

price[100000][3]={0};

memset(cache,-1,sizeof(cache[0][0])\*300000); n=0;

cin>>n;

for(int j=0;j<n;j++) for(int k=0;k<3;k++)

cin>>price[j][k];

int minMoney = min(costOfBuying(0,0), min(costOfBuying(0,1),costOfBuying(0,2)));

cout<<minMoney<<endl;

}

return 0;

}

**Stack of Bricks**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int n, a[10], i;

scanf("%d",&n); for(i=0; i<n; i++)

scanf("%d",&a[i]); int ans = 1000; for(i=0; i<n; i++)

if(a[i]==999 && a[i+1]==1) ans++;

else if(a[i]==999 && a[i-1]==1) ans--;

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

}

return 0;

}

**STOCK MAXIMIZE**

#include<bits/stdc++.h> using namespace std;

int main()

{

int t; scanf("%d",&t); while(t--)

{

int n, a[10], i, idx, b[10], sum = 0; scanf("%d",&n);

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

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

b[n] = n;

for(i=n-1; i>0; i--)

{

if(a[idx]<a[i]) idx = i;

b[i] = idx;

}

for(i=1; i<=n; i++) if((a[b[i]]-a[i])>=0)

sum += (a[b[i]] - a[i]); cout << sum << endl;

}

return 0;

}

**STUDIOUS LITTLE JHOOL**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int n, ans = -1, x, y;

scanf("%d",&n); for(x=0; x<n; x++)

for(y=0; y<n; y++) if((12\*x + 10\*y) == n)

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

}

return 0;

}

**SUPER TWO LETTER STRINGS**

#include<bits/stdc++.h> using namespace std; #define mod 1000000007 int dp[1000][2];

int solve(int N , int P)

{

dp[1][0] = 1;

dp[1][1] = 0;

for(int i=2; i<=N; i++)

{

dp[i][0] = dp[i][1] = 0;

dp[i][0] = (dp[i-1][1]+dp[i-1][0])%mod; for(int j=1; j<P; j++)

{

if(i-j<=0) break; dp[i][1] += dp[i-j][0];

if(dp[i][1] >= mod) dp[i][1] -= mod;

}

}

return ((dp[N][0]+dp[N][1])%mod);

}

int main()

{

int t; cin>>t; while(t--)

{

int n,p; cin>>n>>p;

cout<<solve(n,p)<<endl;

}

return 0;

}

**TABLETS**

#include <stdio.h> int main()

{

int j,dif,n,a[10],med[10]={0},x,sum=0,i; scanf("%d",&n);

x=0; a[0]=0;

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

{

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

dif=a[i]-a[i-1]; if(dif > 0)

med[i]=med[i-1]+1; else if(dif == 0)

med[i]=1; else

{

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

{

med[i]=1;

for( j=i-1; j>=1; j--)

{

if(a[j]-a[j+1] >0 && med[j]<=med[j+1])

{

sum+=1; med[j]=med[j+1]+1;

}

else break;

}

}

else

med[i]=1;

}

sum+=med[i];

}

printf("%d",sum); return 0;

}

**THE COIN EXCHANGE PROBLEM**

#include <stdio.h>

int count(int arr[], int m, int n)

{

if (n == 0) return 1;

if (n < 0 || m <= 0) return 0;

return count(arr,m-1,n) + count(arr,m,n-arr[m-1]);

}

int main()

{

int n , m , a[10] , i;

scanf("%d%d",&n,&m); for(i=0; i<n; i++)

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

printf("%d",count(a,m,n));

return 0;

}

**The colorful street**

#include<bits/stdc++.h> using namespace std; #define ll long long

int main()

{

int t;

cin >> t; while(t--)

{

int n;

cin >> n;

int a[n][3], ans;

cin >> a[0][0] >> a[0][1] >> a[0][2];

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

{

cin >> a[i][0] >> a[i][1] >> a[i][2];

a[i][0] += min(a[i-1][1],a[i-1][2]);

a[i][1] += min(a[i-1][0],a[i-1][2]);

a[i][2] += min(a[i-1][0],a[i-1][1]);

}

ans = min(a[n-1][0],a[n-1][1]); ans = min(a[n-1][2],ans);

cout << ans << "\n";

}

return 0;

}

**The largest possible!**

#include <stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int n, a[10], i, ans = 0;

scanf("%d",&n);

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

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

ans += abs(a[i]-a[i+1]);

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

}

return 0;

}

**THE MAXIMUM SUB ARRAY**

#include <iostream> using namespace std; int main()

{

int t; cin>>t; while(t--)

{

int n , a[10] , i , j; cin>>n;

for(i=0; i<n; i++) cin>>a[i];

int maxc = 0 , maxn = 0; for(i=0; i<n-1; i++)

{

int temp = a[i]; for(j=i+1; j<n; j++)

{

temp += a[j];

if ( temp > maxc ) maxc = temp;

}

}

for(i=0; i<n; i++) if (a[i] > 0)

maxn += a[i];

cout << maxc << " " << maxn << endl;

}

return 0;

}

**TOWER OF HANOI**

#include<bits/stdc++.h> using namespace std;

#define all(x) (x).begin(),(x).end() #define F first

#define S second

bool comp(pair<int,int> a,pair<int,int> b)

{

if(a.F>b.F) return 0; if(a.F<b.F) return 1;

return (a.S<b.S);

}

int main()

{

int T; cin>>T; while(T--)

{

int n; cin>>n;

vector<pair<int,int>> vp; for(int i = 0;i<n;i++)

{

int x,y; cin>>x>>y;

vp.push\_back({x,y});

}

sort(all(vp),comp);

int ans[n],answer = 0;; for(int i = 0;i<n;i++)

{

ans[i] = vp[i].S; int add = 0;

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

if(vp[j].S<vp[i].S and vp[j].F != vp[i].F) add = max(add,ans[j]);

ans[i] += add;

answer = max(answer,ans[i]);

}

cout<<answer<<"\n";

}

return 0;

}

**Travelling Sales Man**

#include<stdio.h>

int matrix[25][25], visited\_cities[10], limit, cost = 0; int tsp(int c)

{

int count, nearest\_city = 999; int minimum = 999, temp;

for(count = 0; count < limit; count++)

if((matrix[c][count] != 0) && (visited\_cities[count] == 0))

{

if(matrix[c][count] < minimum)

minimum = matrix[count][0] + matrix[c][count]; temp = matrix[c][count];

nearest\_city = count;

}

if(minimum != 999) cost = cost + temp;

return nearest\_city;

}

void minimum\_cost(int city)

{

int nearest\_city; visited\_cities[city] = 1; printf("%d ", city + 1); nearest\_city = tsp(city); if(nearest\_city == 999)

{

nearest\_city = 0;

printf("%d", nearest\_city + 1);

cost = cost + matrix[city][nearest\_city]; return;

}

minimum\_cost(nearest\_city);

}

int main()

{

int i, j;

scanf("%d", &limit); for(i = 0; i < limit; i++)

{

for(j = 0; j < limit; j++) scanf("%d", &matrix[i][j]);

visited\_cities[i] = 0;

}

printf("Path: "); minimum\_cost(0); printf("\nMinimum Cost: "); printf("%d\n", cost);

return 0;

}

**Circles**

#include<stdio.h> int G[50][50],x[50];

void next\_color(int k){ int i,j;

x[k]=1;

for(i=0;i<k;i++){ if(G[i][k]!=0 && x[k]==x[i])

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

**DAA E-Lab Programs Session-6: Backtracking**

}

}

int main(){ int n,e,i,j,k,l;

scanf("%d",&n);

scanf("%d",&e); for(i=0;i<n;i++)

for(j=0;j<n;j++) G[i][j]=0;

for(i=0;i<e;i++){ scanf("%d %d",&k,&l);

G[k][l]=1;

G[l][k]=1;

}

for(i=0;i<n;i++) next\_color(i);

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

{

if(x[i]==1) printf("Vertex[%d] : 1\n",i+1);

if(x[i]==2)

printf("Vertex[%d] : 2\n",i+1); if(x[i]==3)

printf("Vertex[%d] : 3\n",i+1);

}

return 0;

}

#### Colors

#include<stdio.h> int G[50][50],x[50];

void next\_color(int k){ int i,j;

x[k]=1;

for(i=0;i<k;i++){ if(G[i][k]!=0 && x[k]==x[i])

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

}

}

int main(){ int n,e,i,j,k,l;

scanf("%d",&n);

scanf("%d",&e); for(i=0;i<n;i++)

for(j=0;j<n;j++) G[i][j]=0;

for(i=0;i<e;i++){ scanf("%d %d",&k,&l);

G[k][l]=1;

G[l][k]=1;

}

for(i=0;i<n;i++) next\_color(i);

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

{

if(x[i]==1)

printf("Vertex[%d] : Green\n",i+1);

if(x[i]==2)

printf("Vertex[%d] : Yellow\n",i+1); if(x[i]==3)

printf("Vertex[%d] : Red\n",i+1);

}

return 0;

}

**Students** #include<stdio.h> int G[50][50],x[50];

void next\_color(int k){ int i,j;

x[k]=1;

for(i=0;i<k;i++){ if(G[i][k]!=0 && x[k]==x[i])

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

}

}

int main(){ int n,e,i,j,k,l;

scanf("%d",&n);

scanf("%d",&e); for(i=0;i<n;i++)

for(j=0;j<n;j++) G[i][j]=0;

for(i=0;i<e;i++){ scanf("%d %d",&k,&l);

G[k][l]=1;

G[l][k]=1;

}

for(i=0;i<n;i++) next\_color(i);

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

{

if(x[i]==1)

printf("Vertex[%d] : Red\n",i+1);

if(x[i]==2)

printf("Vertex[%d] : Blue\n",i+1); if(x[i]==3)

printf("Vertex[%d] : Yellow\n",i+1);

}

return 0;

}

#### Graph 1

#include<stdio.h>

int a[20][20],reach[20],n; void dfs(int v){

int i; reach[v]=1;

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

if(a[v][i]&&!reach[i]){

printf("\n%d->%d",v,i); dfs(i);

}

}

int main(){

int i,j,count=0;

scanf("%d",&n); for(i=1;i<=n;i++)

for(j=1;j<=n;j++){ reach[i]=0; a[i][j]=0;

}

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

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

if(reach[i]) count++; if(count==n)

printf("\nGraph is connected"); else

printf("\nGraph is disconnected");

return(0);

}

#### Reverse Delete Algorithm for Minimum Spanning Tree

#include<iostream> #include <bits/stdc++.h> using namespace std;

int root(int a[],int x){ while(a[x]!=x){

a[x]=a[a[x]]; x=a[x];

}

return x;

}

bool f(pair<int,pair<int,int> > p1,pair<int,pair<int,int> > p2){ return p1.first<p2.first;

}

int main()

{

//code int t; cin>>t;

while(t--){ int n,e;

cin>>n>>e; pair<int,pair<int,int> > a[e]; for(int i=0;i<e;i++){

int x,y,d; cin>>x>>y>>d;

a[i]=make\_pair(d,make\_pair(x,y));

}

sort(a,a+e,f); int b[n];

for(int i=0;i<n;i++){ b[i]=i;

}

int ans=0;

for(int i=0;i<e;i++){ pair<int,pair<int,int> > p=a[i]; int d=p.first;

int x=p.second.first,y=p.second.second; if(root(b,x)!=root(b,y)){

b[root(b,x)]=b[root(b,y)]; ans+=d;

}

}

cout<<ans<<endl;

}

return 0;

}

#### Blank cells

#include <bits/stdc++.h> using namespace std; struct cell

{

int x,y; cell(){} cell(int a,int b)

{x=a,y=b;}

};

int valid(int x,int y,int n)

{

return (x>0 && x<=n && y>0 && y<=n);

}

int bfs(int n)

{

int g[n+1][n+1]; int s[2],d[2],x,y;

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

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

{

cin>>g[i][j];

if(g[i][j]==1)

s[0]=i,s[1]=j;

else if(g[i][j]==2)

d[0]=i,d[1]=j;

}

queue<cell> q;

cell source=cell(s[0],s[1]); q.push(source);

bool visited[n+1][n+1]; for(int i=1;i<=n;i++) for(int j=1;j<=1;j++) visited[i][j]=0;

visited[source.x][source.y]=1; cell t;

while(!q.empty())

{

int dx[4]={1,-1,0,0};

int dy[4]={0,0,1,-1};

t=q.front(),q.pop();

if(t.x==d[0] && t.y==d[1]) return 1;

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

{

x=t.x+dx[i];

y=t.y+dy[i];

if((g[x][y]==3 || g[x][y]==2) && valid(x,y,n) && !visited[x][y]) q.push(cell(x,y)),visited[x][y]=1;

}

}

return 0;

}

int main()

{

int t,n; cin>>t; while(t--)

{

cin>>n; cout<<bfs(n)<<endl;

}

}

**Distance** #include<iostream> #include <vector> #include<stack> #include<queue> using namespace std; int main()

{

int t; cin>>t;

while(t--)

{

int m,n; cin>>m>>n; int mat[m][n];

for (int i=0;i<m;i++) for (int j=0;j<n;j++)

cin>>mat[i][j];

queue<pair<int,int> > q; for (int i=0;i<m;i++)

for (int j=0;j<n;j++) if (mat[i][j])

{

q.push(make\_pair(i,j));

}

while(!q.empty())

{

pair<int,int> current = q.front(); q.pop();

int x = current.first,y=current.second; if (y+1<n && mat[x][y+1] == 0)

mat[x][y+1] = mat[x][y]+1,q.push(make\_pair(x,y+1));

if (y-1>=0 && mat[x][y-1] == 0)

mat[x][y-1] = mat[x][y]+1,q.push(make\_pair(x,y-1));;

if (x+1<m && mat[x+1][y] == 0)

mat[x+1][y] = mat[x][y]+1,q.push(make\_pair(x+1,y));;

if (x-1>=0 && mat[x-1][y] == 0)

mat[x-1][y] = mat[x][y]+1,q.push(make\_pair(x-1,y));;

}

for (int i=0;i<m;i++) for(int j=0;j<n;j++) cout<<mat[i][j]-1<<" "; cout<<endl;

}

return 0;

}

**Word Boggle** #include<stdio.h> #include<string.h> #include<stdlib.h>

int strcmpfunc(const void \*a, const void \*b)

{

return (strcmp((char \*)a, (char \*)b));

}

int main()

{

int tst; scanf("%d",&tst);

while(tst--)

{

int strings; scanf("%d",&strings);

char str[strings][20]; int i;

for(i = 0; i<strings; i++) scanf(" %s",str[i]);

/\*Sort the array.\*/

qsort(str, strings, sizeof(char)\*20, strcmpfunc);

int pre\_index = 0;

/\*Remove duplicate\*/ for(i = 1; i<strings; i++)

{

if (!strcmp(str[i], str[pre\_index])) str[i][0] = '\0';

else

pre\_index = i;

}

int ch[82], str\_ch[82], no\_word = 1;

for(i = 0; i<82; i++)

{

ch[i] = 0;

str\_ch[i] = 0;

}

int x, y, j; scanf("%d", &x);

scanf("%d", &y);

for(j = 0; j<x\*y; j++)

{

char input;

scanf(" %c", &input);

ch[input-'A']++;

}

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

{

if (!strlen(str[i])) continue;

for(j = 0; j<82; j++) str\_ch[j] = 0;

for (j = 0; j<strlen(str[i]); j++) str\_ch[str[i][j]-'A']++;

for(j = 0; j<82; j++)

if (str\_ch[j] && str\_ch[j] > ch[j]) break;

if (j == 82)

{

printf("%s ",str[i]); no\_word = 0;

}

}

if (no\_word) printf("-1");

printf("\n");

}

return 0;

}

#### Knight Walk

#include <bits/stdc++.h> #include <queue> #include <vector>

using namespace std;

struct pos{

pos(int x0,int y0,int move0){ x=x0;

y=y0; move=move0;

}

int x; int y;

int move; bool vis;

};

int main(){ int T; cin>>T;

for (int i =0;i<T;i++){ int N,M;

cin>>N; cin>>M;

int s1,s2,d1,d2; cin>>s1>>s2>>d1>>d2; bool vis[N][M];

int xpos[]= {2,2,-2,-2,1,1,-1,-1};

int ypos[]= {1,-1,1,-1,2,-2,2,-2};

for (int j=0;j<N;j++){

for (int k=0;k<M;k++){ vis[j][k]=false;

}

}

queue<pos> q;

pos p0(s1-1,s2-1,0); q.push(p0);

int count=0;

while (!q.empty()){ pos cur=q.front(); q.pop();

if (cur.x==d1-1&&cur.y==d2-1) { cout<< cur.move<<endl; count=1;

break;

}

for (int j=0;j<8;j++){

int xNew=cur.x+xpos[j]; int yNew=cur.y+ypos[j];

if(vis[xNew][yNew]||xNew<0||xNew>=N||yNew<0||yNew>=M) continue; vis[xNew][yNew]=true;

pos pTemp(xNew,yNew,cur.move+1); q.push(pTemp);

}

}

if (count==0) cout<<-1<<endl;

}

return 0;

}

**Replace O's with X's** #include <iostream> using namespace std; int m,n;

void fill(char mat[10][10],int x,int y,char prev,char newv)

{

if(x<0||x>=m||y<0||y>=n) return ; if(mat[x][y]!=prev) return; mat[x][y]=newv; fill(mat,x+1,y,prev,newv); fill(mat,x-1,y,prev,newv); fill(mat,x,y+1,prev,newv); fill(mat,x,y-1,prev,newv);

}

int replace(char mat[10][10])

{

int i,j; for(i=0;i<m;i++)

for(j=0;j<n;j++) if(mat[i][j]=='O')

mat[i][j]='-'; for(i=0;i<m;i++)

if(mat[i][0]=='-')

fill(mat,i,0,'-','O'); for(i=0;i<m;i++)

if(mat[i][n-1]=='-')

fill(mat,i,n-1,'-','O'); for(i=0;i<n;i++)

if(mat[0][i]=='-')

fill(mat,0,i,'-','O'); for(i=0;i<n;i++)

if(mat[m-1][i]=='-')

fill(mat,m-1,i,'-','O'); for(i=0;i<m;i++)

for(j=0;j<n;j++) if(mat[i][j]=='-')

mat[i][j]='X';

}

int main()

{

int t,i,j; cin>>t;

char mat[10][10];

while(t--)

{

cin>>m>>n; for(i=0;i<m;i++)

for(j=0;j<n;j++) cin>>mat[i][j];

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

for(j=0;j<n;j++) cout<<mat[i][j]<<" ";

cout<<endl;

}

return 0;

}

#### Flood fill Algorithm

#include <iostream> #include <queue> using namespace std;

int main() {

int t; cin>>t; while(t--)

{

int n,m,x,y,k; cin>>n>>m;

vector<vector<int>> v(n,vector<int>(m,0)); vector<vector<bool>> visited(n,vector<bool>(m,false)); for(int i=0;i<n;i++)

for(int j=0;j<m;j++) cin>>v[i][j];

cin>>x>>y>>k; int color=v[x][y];

queue<pair<int,int>> q; q.push(make\_pair(x,y));

while(!q.empty())

{

pair<int,int> p=q.front(); q.pop();

v[p.first][p.second]=k; visited[p.first][p.second]=true;

if(p.first+1<n && !visited[p.first+1][p.second] && v[p.first+1][p.second]==color) q.push(make\_pair(p.first+1,p.second));

if(p.first-1>=0 && !visited[p.first-1][p.second] && v[p.first-1][p.second]==color) q.push(make\_pair(p.first-1,p.second));

if(p.second+1<m && !visited[p.first][p.second+1] && v[p.first][p.second+1]==color) q.push(make\_pair(p.first,p.second+1));

if(p.second-1>=0 && !visited[p.first][p.second-1] && v[p.first][p.second-1]==color) q.push(make\_pair(p.first,p.second-1));

}

for(int i=0;i<n;i++) for(int j=0;j<m;j++)

cout<<v[i][j]<<" "; cout<<endl;

}

return 0;

}

**Divisibility tree** #include<bits/stdc++.h> using namespace std;

int ans;

int dfs(vector<int> arr[1001],int N,int node=1)

{

int total=1;

for(auto it=arr[node].begin();it!=arr[node].end();it++)

{

int nodes\_below=dfs(arr,N,\*it); total+=nodes\_below; if(nodes\_below%2==0)

ans++;

}

return total;

}

int main()

{

int T; cin>>T; while(T--)

{

int N,M; cin>>N>>M; ans=0;

vector<int> arr[1001]; for(int i=0;i<M;i++)

{

int x,y; cin>>x>>y;

arr[y].push\_back(x);

}

dfs(arr,N); cout<<ans<<endl;

}

return 0;

}

#### Shortest Source to Destination Path

#include<bits/stdc++.h> using namespace std; #define ll long long

int mat[200][200]; int vis[200][200]; int m,n;

bool is(int i,int j){

if( (i >= 0) && (j >= 0) &&(i < m) && (j < n )&& (mat[i][j] == 1)){ return true;

}

return false;

}

int main(){

int i,j,k,l,T,x,y,p,q,d; cin>>T;

while(T--){

queue< pair < int ,pair<int,int> > > pq; pair < int ,pair<int,int> > pr; cin>>m>>n;

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

scanf("%d",&mat[i][j]);vis[i][j]=0;

}

}

cin>>x>>y; pq.push(make\_pair(0,make\_pair(0,0))); int fg=0;

while(!pq.empty()){ pr= pq.front(); pq.pop();

d = pr.first;

p = pr.second.first;q=pr.second.second; if((pr.second.first == x) && (pr.second.second == y )){

fg=1;break;

}

if(vis[p][q] == 0){vis[p][q]=1;

if(is(p+1,q)){

pr =make\_pair(d+1,make\_pair(p+1,q)); pq.push(pr);

}

if(is(p-1,q)){

pr =make\_pair(d+1,make\_pair(p-1,q));pq.push(pr);

}

if(is(p,q+1)){

pr =make\_pair(d+1,make\_pair(p,q+1));pq.push(pr);

}

if(is(p,q-1)){

pr =make\_pair(d+1,make\_pair(p,q-1));pq.push(pr);

}

}

}

if(mat[0][0]==0||mat[x][y]==0)fg=0; if(fg)

printf("%d\n",d);else printf("-1\n");

}

return 0;

}

**Hamiltonian Path** #include <stdio.h> int n;

int checkhamiltonian(int a[n][n],int v,int count,int visited[n])

{

if (count == n)

{

return 1;

}

int i; visited[v] = 1;

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

{

if (a[v][i] == 1 && visited[i] == 0)

{

visited[i] = 1;

if (checkhamiltonian(a,i,count + 1,visited))

{

return 1;

}

visited[i] = 0;

}

} return 0;

}

int main()

{

int t,i,j,m,x,y;

scanf("%d", &t); while (t > 0)

{

t--;

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

int a[n][n],visited[n]; for (i = 0; i < n; i++)

{

visited[i] = 0;

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

{

a[i][j] = 0; }

}

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

{ scanf("%d%d", &x, &y); x--;

y--;

a[x][y] = 1;

a[y][x] = 1;

} int result;

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

{

result = checkhamiltonian(a,i,1,visited); for (j = 0; j < n; j++)

{ visited[j] = 0; } if (result == 1)

{

break;

}

} printf("%d\n", result);

}

return 0;

}

**Your Social Network** #include <iostream> using namespace std;

int main() {

int t,n,i,j; cin>>t; while(t--)

{

cin>>n; int frien[n];

int relation[n][n]; for(i=1;i<n;i++)

{

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

{

relation[i][j]=-1;

}

}

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

{

cin>>frien[i]; frien[i]--;

}

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

{

relation[i][frien[i]]=1; if(i!=1)

{

for(j=0;j<frien[i];j++)

{

if(relation[frien[i]][j]!=-1)

{

relation[i][j] = relation[frien[i]][j] + 1;

}

}

}

}

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

{

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

{

if(relation[i][j]!=-1)

{

cout<<(i+1)<<" "<<(j+1)<<" "<<relation[i][j]<<" ";

}

}

}

cout<<endl;

}

return 0;

}

#### Possible paths

#include <stdio.h>

int count(int g[25][25], int u, int v, int k, int N, int dp[100][100][100])

{

if(k <= 0)

return 0;

if(u == v && k == 0) return 1;

if((k == 1) && g[u][v] == 1) return 1;

if(dp[u][v][k] != -1) return dp[u][v][k];

int i;

int sum = 0;

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

{

if(g[u][i] == 1)

sum += count(g, i, v, k - 1, N, dp);

}

dp[u][v][k] = sum;

return dp[u][v][k];

}

int main() {

//code int T;

scanf("%d", &T); while(T--)

{

int N; scanf("%d", &N);

int g[25][25]; int i, j, k;

int dp[100][100][100];

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

{

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

{

scanf("%d", &g[i][j]); for(k = 0; k < 100; k++)

{

dp[i][j][k] = -1;

}

}

}

int u, v;

scanf("%d %d %d", &u, &v, &k);

int ans = count(g, u, v, k, N, dp); printf("%d\n", ans);

}

return 0;

}

**Nodes at even distance** #include <iostream> #include <vector> #include <algorithm> using namespace std;

void dfs(int pos,vector<vector<int> > &v,vector<int> &vis){ if(pos>=vis.size()) return;

//cout<<pos<<" -> " <<v[pos].size()<<endl; for(int i=0;i<v[pos].size();i++){

//cout<<vis[v[pos][i]]<<endl; if(vis[v[pos][i]] == 0){

vis[v[pos][i]] = vis[pos]+1;

dfs(v[pos][i],v,vis);

}

}

}

int main() {

int t; cin>>t; while(t--){

int n; cin>>n;

vector<vector<int> > v(n+1); for(int i= 0;i<n-1;i++){

int p,q; cin>>p>>q; v[p].push\_back(q);

v[q].push\_back(p);

}

vector<int> vis(n+1,0); vis[1] = 1;

dfs(1,v,vis); int cnt1 = 0; int cnt2 = 0;

for(int i=1;i<vis.size();i++){

//cout<<vis[i]<<" "; if(vis[i]%2==0)cnt1++; if(vis[i]%2!=0)cnt2++;

}

//cout<<endl; int sm = 0; if(cnt1>1){

sm+=cnt1\*(cnt1-1)/2;

}

if(cnt2>1){ sm+=cnt2\*(cnt2-1)/2;

}

cout<<sm<<endl;

}

return 0;

}

**Circle of strings** #include<bits/stdc++.h> using namespace std;

bool dfs(vector<int> arr[100],bool mark[100],int N,int node,int first\_node)

{

if(node!=first\_node) mark[node]=1;

for(auto it=arr[node].begin();it!=arr[node].end();it++) if(!mark[\*it] && dfs(arr,mark,N,\*it,-1))

return 1;

for(int i=0;i<N;i++) if(!mark[i])

{

mark[node]=0; return 0;

}

mark[node]=0;

return 1;

}

int main()

{

int T; cin>>T; while(T--)

{

int N; cin>>N; string str[N];

for(int i=0;i<N;i++) cin>>str[i];

vector<int> arr[100]; for(int i=0;i<N;i++)

{

int l=str[i].size()-1; for(int j=0;j<N;j++)

if(i!=j && str[i][l]==str[j][0]) arr[i].push\_back(j);

}

if(N==1 && str[0][0]==str[0][str[0].size()-1])

{

cout<<"1"<<endl; goto label;

}

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

{

bool mark[100]={0};

if(dfs(arr,mark,N,i,i))

{

cout<<"1"<<endl; goto label;

}

}

cout<<"0"<<endl; label:;

}

}

**Connecting Nodes graph** #include <iostream> #include<vector> #include<stack> #include<algorithm> using namespace std;

int main()

{

int t,i,n,m,u,v; cin>>t; while(t--)

{

cin>>n>>m;

vector <int>graph[n+1];

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

{

cin>>u>>v; graph[u].push\_back(v); graph[v].push\_back(u);

}

bool visited[n+1]; fill(visited,visited+n+1,false); int he=0;

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

{

if(visited[i]) continue; int ov=0; stack<int>st; st.push(i);

while(!st.empty())

{

int vv=st.top(); st.pop(); if(visited[vv])

continue; visited[vv]=true; if(graph[vv].size()%2==1)

{ ++ov;}

for(vector<int>::iterator it=graph[vv].begin();it!=graph[vv].end();++it) st.push(\*it);

}

he=he+ov; if(ov==0)he=he+2;

}

cout<<he/2<<endl;

}

return 0;

}

#### Path of greater than equal to k length

#include <stdio.h> int grp[50][50],

visi[50];int dfs(int sum,int k,int v,int V)

{

int i;

if(sum>=k) return 1; for(i=0;i<V;i++)

{

if(grp[v][i]!=0 && visi[i]==0)

{

visi[i]=1; if(dfs(sum+grp[v][i],k,i,V)==1)

return 1; visi[i]=0;

}

} return 0;

}

int main()

{

int t;scanf("%d",&t); while(t--)

{

int i,j,e,v,k,s,t,w;

scanf("%d %d %d",&v,&e,&k); for(i=0;i<v;i++)

{

for(j=0;j<v;j++) grp[i][j]=0;

}

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

{

scanf("%d %d %d",&s,&t,&w); grp[s][t]=w;

grp[t][s]=w;

}

for(i=0;i<v;i++) visi[i]=0; visi[0]=1; printf("%d\n",dfs(0,k,0,v));

}

return 0;

}

**Alphabets** #include <stdio.h> #include <string.h>

void swap(char \*x, char \*y)

{

char temp; temp = \*x;

\*x = \*y;

\*y = temp;

}

void permute(char \*a, int l, int r)

{

int i;

if (l == r) printf("%s\n", a);

else

{

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

{

swap((a+l), (a+i)); permute(a, l+1, r); swap((a+l), (a+i));

}

}

}

int main()

{

char str[10]; scanf("%s",str);

int n = strlen(str); permute(str, 0, n-1);

return 0;

}

#### Numbers

#include <stdio.h> #include <string.h>

void swap(char \*x, char \*y)

{

char temp; temp = \*x;

\*x = \*y;

\*y = temp;

}

void permute(char \*a, int l, int r)

{

int i;

if (l == r)

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

else

{

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

{

swap((a+l), (a+i)); permute(a, l+1, r); swap((a+l), (a+i));

}

}

}

int main()

{

char str[30] ; scanf("%s",str); int n = strlen(str);

permute(str, 0, n-1);

return 0;

}

**Permutations** #include <stdio.h> #include <string.h>

void swap(char \*x, char \*y)

{

char temp; temp = \*x;

\*x = \*y;

\*y = temp;

}

void permute(char \*a, int l, int r)

{

int i;

if (l == r)

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

else

{

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

{

swap((a+l), (a+i)); permute(a, l+1, r); swap((a+l), (a+i));

}

}

}

int main()

{

char str[30] ; scanf("%s",str); int n = strlen(str);

permute(str, 0, n-1);

return 0;

}

**Alphabets 1** #include <stdio.h> #include <string.h>

void swap(char \*x, char \*y)

{

char temp; temp = \*x;

\*x = \*y;

\*y = temp;

}

void permute(char \*a, int l, int r)

{

int i;

if (l == r) printf("%s\n", a);

else

{

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

{

swap((a+l), (a+i)); permute(a, l+1, r); swap((a+l), (a+i));

}

}

}

int main()

{

char str[10]; scanf("%s",str);

int n = strlen(str); permute(str, 0, n-1);

return 0;

}

**Permutations 1** #include <stdio.h> #include <string.h>

void swap(char \*x, char \*y)

{

char temp; temp = \*x;

\*x = \*y;

\*y = temp;

}

void permute(char \*a, int l, int r)

{

int i;

if (l == r) printf("%s\n", a);

else

{

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

{

swap((a+l), (a+i)); permute(a, l+1, r); swap((a+l), (a+i));

}

}

}

int main()

{

char str[100]; scanf("%s",str);

int n = strlen(str); permute(str, 0, n-1);

return 0;

}

#### Array

#include<stdio.h> #include<string.h>

void print(char \*num, int n)

{

int i;

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

printf("\n");

}

int main()

{

char num[10]; scanf("%s",num);

char \*ptr; char temp; int i, n=3, j;

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

for (i = 0; i < n-1; i++) { temp = num[i]; num[i] = num[i+1]; num[i+1] = temp; print(num, n);

}

}

return 0;

}

#### String

#include <stdio.h> #include <string.h>

void swap(char \*x, char \*y)

{

char temp; temp = \*x;

\*x = \*y;

\*y = temp;

}

void permute(char \*a, int l, int r)

{

int i;

if (l == r) printf("%s\n", a);

else

{

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

{

swap((a+l), (a+i)); permute(a, l+1, r); swap((a+l), (a+i));

}

}

}

int main()

{

char str[100]; scanf("%s",str);

int n = strlen(str); permute(str, 0, n-1);

return 0;

}

**Chess** #include<stdio.h> #include<math.h>

int x[50],soln=0;

int place(int k,int i){ int j; for(j=1;j<k;j++)

if((x[j]==i) || (abs(x[j]-i)==abs(j-k))) return 0;

return 1;

}

void display(int n){ int i,j;

soln++;

printf("\nSOLUTION #%d\n",soln); for(i=1;i<=n;i++){

for(j=1;j<=n;j++) if(x[i]==j)

printf("Q "); else

printf("\* ");

printf("\n");

}

}

void nqueens(int k,int n){ int i;

for(i=1;i<=n;i++) if(place(k,i)==1){ x[k]=i;

if(k==n) display(n);

else

nqueens(k+1,n);

}

}

int main(){ int n;

// printf("Enter no. of queens : "); scanf("%d",&n);

nqueens(1,n);

printf("TOTAL SOLN. : %d",soln);

return 0;

}

**Chess 1** #include<stdio.h> #include<math.h>

int x[50],soln=0;

int place(int k,int i){ int j; for(j=1;j<k;j++)

if((x[j]==i) || (abs(x[j]-i)==abs(j-k))) return 0;

return 1;

}

void display(int n){ int i,j;

soln++;

printf("\nSOLUTION #%d\n",soln); for(i=1;i<=n;i++){

for(j=1;j<=n;j++) if(x[i]==j)

printf("Q "); else

printf("\* ");

printf("\n");

}

}

void nqueens(int k,int n){ int i;

for(i=1;i<=n;i++) if(place(k,i)==1){ x[k]=i;

if(k==n) display(n);

else

nqueens(k+1,n);

}

}

int main(){ int n;

scanf("%d",&n); nqueens(1,n);

printf("TOTAL SOLN. : %d",soln);

return 0;

}

**Chess 2** #include<stdio.h> #include<math.h>

int x[50],soln=0;

int place(int k,int i){ int j; for(j=1;j<k;j++)

if((x[j]==i) || (abs(x[j]-i)==abs(j-k)))

return 0;

return 1;

}

void display(int n){ int i,j;

soln++;

printf("SOLUTION #%d\n",soln); for(i=1;i<=n;i++){

for(j=1;j<=n;j++) if(x[i]==j)

printf("Q "); else

printf("\* ");

printf("\n");

}

}

void nqueens(int k,int n){ int i;

for(i=1;i<=n;i++) if(place(k,i)==1){ x[k]=i;

if(k==n) display(n);

else

nqueens(k+1,n);

}

}

int main(){ int n;

scanf("%d",&n); nqueens(1,n);

printf("TOTAL SOLN. : %d",soln);

return 0;

}

**Chess 3** #include<stdio.h> #include<math.h>

int board[20],count; int cnt=0;

int main()

{

int n,i,j;

void queen(int row,int n); scanf("%d",&n);

queen(1,n);

printf("\nTOTAL SOLN. : %d",cnt); return 0;

}

//function for printing the solution int print(int n)

{

int i,j;

printf("\n\nSOLUTION #%d",++count); cnt=count;

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

{

printf("\n");

for(j=1;j<=n;++j) //for nxn board

{

if(board[i]==j)

printf("Q "); //queen at i,j position

else

printf("\* "); //empty slot

}

printf("\n");

}

return count;

}

/\*funtion to check conflicts

If no conflict for desired postion returns 1 otherwise returns 0\*/ int place(int row,int column)

{

int i;

for(i=1;i<=row-1;++i)

{

//checking column and digonal conflicts if(board[i]==column)

return 0; else

if(abs(board[i]-column)==abs(i-row)) return 0;

}

return 1; //no conflicts

}

//function to check for proper positioning of queen void queen(int row,int n)

{

int column; for(column=1;column<=n;++column)

{

if(place(row,column))

{

board[row]=column; //no conflicts so place queen if(row==n) //dead end

print(n);//printing the board configuration

else //try queen with next position queen(row+1,n);

}

}

}

**Queen** #include<stdio.h> #include<math.h> int a[30],count=0; int place(int pos)

{

int i; for(i=1;i<pos;i++)

{

if((a[i]==a[pos])||((abs(a[i]-a[pos])==abs(i-pos)))) return 0;

}

return 1;

}

void print\_sol(int n)

{

int i,j; count++;

printf("SOLUTION #%d\n",count); for(i=1;i<=n;i++)

{

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

{

if(a[i]==j)

printf("Q "); else printf("\* ");

}

printf("\n");

}

}

void queen(int n)

{

int k=1; a[k]=0;

while(k!=0)

{

a[k]=a[k]+1; while((a[k]<=n)&&!place(k)) a[k]++;

if(a[k]<=n)

{

if(k==n) print\_sol(n); else

{

k++; a[k]=0;

}

}

else k--;

}

}

int main()

{

int i,n; scanf("%d",&n); queen(n);

printf("TOTAL SOLN. : %d",count); return 0;

}

# DAA E-Lab

**Session-7**

#### Monk at the Graph Factory

#include <stdio.h>

int main()

{

int n;

scanf("%d", &n); int sumdeg = 0; int i;

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

{

int k; scanf("%d", &k); sumdeg += k;

}

puts(sumdeg == 2\*(n-1) ? "Yes" : "No");

return 0;

}

**Monk in the real estate** #include<iostream> #include<vector> #include<algorithm> using namespace std; int main()

{

int t; cin>>t ;

for(int tt=0;tt<t;tt++)

{ int c=0; int n;

cin>>n; int x,y; int A[10010]={0};

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

{cin>>x>>y;

if(A[x]==0)

{A[x]++; c++;} if(A[y]==0)

{A[y]++;

c++;

}

}

cout<<c<<endl;

}

return 0;

}

**Monk Learning Graph** #include <stdio.h> #include <stdlib.h>

#define N 1000

struct element { int a, i;

} aa[N][N];

int compare(const void \*a, const void \*b) { struct element \*ia = (struct element \*) a; struct element \*ib = (struct element \*) b;

return ib->a != ia->a ? ib->a - ia->a : ib->i - ia->i;

}

int main() { int i, n, m, k;

static int val[N], cnt[N];

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

for (i = 0; i < n; i++) scanf("%d", &val[i]); while (m-- > 0) {

int x, y;

scanf("%d%d", &x, &y);

x--, y--;

aa[x][cnt[x]].a = val[y];

aa[x][cnt[x]++].i = y;

aa[y][cnt[y]].a = val[x];

aa[y][cnt[y]++].i = x;

}

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

qsort(aa[i], cnt[i], sizeof \*aa[i], compare); for (i = 0; i < n; i++)

printf("%d\n", k >= cnt[i] ? -1 : aa[i][k].i + 1);

return 0;

}

#### BFS-Oliver and the battle

#include <stdio.h> #define MAXN 1010

int grid[MAXN][MAXN]; int foi[MAXN][MAXN];

int queue[MAXN\*MAXN][2];

int dx[] = {0,1,1,1,0,-1,-1,-1};

int dy[] = {-1,-1,0,1,1,1,0,-1};

int n,m;

int bfs(int row,int col) { int l = 0, r = 0; queue[r][0] = row;

queue[r][1] = col;

foi[row][col] = 1; r++;

int ret = 1; while(l < r) {

int cr = queue[l][0], cc = queue[l][1]; l++;

int i;

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

int nr = cr + dx[i], nc = cc + dy[i];

if(nr < 0 || nc < 0 || nr >= n || nc >= m) continue; if(!foi[nr][nc] && grid[nr][nc] == 1) {

foi[nr][nc] = 1; ret++; queue[r][0] = nr;

queue[r][1] = nc; r++;

}

}

}

return ret;

}

int main() { int t;

for(scanf("%d",&t); t--;) { scanf("%d%d",&n,&m); int a,b;

for(a=0;a<n;++a) { for(b=0;b<m;++b) { scanf("%d",&grid[a][b]); foi[a][b] = 0;

}

}

int cnt = 0, max = 0; for(a=0; a<n; ++a) { for(b=0; b<m; ++b) {

if(!foi[a][b] && grid[a][b] == 1) { cnt++;

int got = bfs(a,b);

if(got > max) max = got;

}

}

}

printf("%d %d\n",cnt,max);

}

return 0;

}

**BFS-Agitated Chandan** #include <bits/stdc++.h> using namespace std;

int dist[100],visited[100],n,tme; struct gr

{

int i,d;

};

typedef struct gr node; vector<node> v[100]; int bfs(int start)

{

queue<int>q; q.push(start); for(int i=0;i<=n;i++)

{

visited[i]=0; dist[i]=0;

}

visited[start]=1; while(!q.empty())

{

int temp=q.front(); q.pop();

for(int i=0;i<(int)v[temp].size();i++)

{

if(visited[v[temp][i].i]==0)

{

visited[v[temp][i].i]=1; dist[v[temp][i].i]+=dist[temp]+v[temp][i].d; q.push(v[temp][i].i);

}

}

}

return int(max\_element(dist+1,dist+n+1)-dist);

}

int main()

{

int u1,v1,d; node temp; int tc; cin>>tc; while(tc--)

{

cin>>n;

for(int i=1;i<=n;i++) v[i].clear();

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

{

cin>>u1>>v1>>d; temp.i=v1; temp.d=d;

v[u1].push\_back(temp); temp.i=u1; v[v1].push\_back(temp);

}

int start=bfs(1); int ans=bfs(start); int f=dist[ans];

int m=0; if(f>100) m=100;

if(f>1000) m=1000; if(f>10000) m=10000;

cout<<m<<" "<<f<<endl;

}

return 0;

}

**DFS-Bishu and his Girlfriend** #include<iostream> #include<vector>

using namespace std;

vector<int>arr[1001]; int visited[1001]={0};

void dfs(int s,int length)

{

visited[s]=length;

for(int i=0;i<arr[s].size();i++)

{

if(visited[arr[s][i]]==0)

dfs(arr[s][i],length+1);

}

}

main()

{

int n; cin>>n;

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

{

int x,y; cin>>x>>y;

arr[x].push\_back(y); arr[y].push\_back(x);

}

int l=1; dfs(1,l); int q; cin>>q;

int mi=99999; int mn=0; while(q-->0)

{

int num; cin>>num; if(mi>visited[num])

{

mi=visited[num]; mn=num;

}

}

cout<<mn<<endl;

}

**DFS-Happy Vertices** #include<iostream> #include<bits/stdc++.h> using namespace std;

vector<int> adj[100001]; bool vis[100001];

int c;

stack <int> stck; void dfs(int s)

{

int p; stck.push(s); vis[s]=true;

while(!stck.empty())

{

p=stck.top();

stck.pop();

for(int i=0;i<adj[p].size();i++)

{

if(vis[adj[p][i]]==false)

{

stck.push(adj[p][i]);

vis[adj[p][i]]=true; if((adj[adj[p][i]].size())-1>adj[p].size())

{

c++;

}

else if((adj[adj[p][i]].size())-1>adj[p].size()-1)

{

c++;

}

}

}

}

}

int main()

{

int n,m,x,y;

scanf("%d %d",&n,&m); for(int i=1;i<=m;i++)

{

scanf("%d %d",&x,&y); adj[x].push\_back(y); adj[y].push\_back(x);

}

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

{

if(vis[i]==false)

dfs(i);

}

cout << c <<"\n";

}

#### DFS-Easy Life

#include <bits/stdc++.h> using namespace std;

int vis[100009]; int sp[100009];

int main() {

int n, m; scanf("%d%d", &n, &m); vector <int> v[100009];

while(m--)

{

int x, y; scanf("%d%d", &x, &y);

v[x].push\_back(y);

v[y].push\_back(x);

}

float max = -1, e2, v2; for(int i=1; i<=n; i++)

{

if(vis[i] == 0)

{

queue <int> q; q.push(i);

vis[i] = 1;

//int count = 0; float e1 = 0, v1 = 0; while(!q.empty())

{

int p = q.front(); q.pop();

v1++;

for(int i=0; i<v[p].size(); i++)

{

if(sp[v[p][i]] == 0)

{

e1 ++;

}

}

sp[p] = 1;

for(int i=0; i<v[p].size(); i++)

{

if(vis[v[p][i]] == 0)

{

vis[v[p][i]] = 1;

q.push(v[p][i]);

}

}

}

float den = e1/v1;

//cout << e1 << " " << v1 << " " << den << endl;

//if(count > max ) max = count;

if(den > max)

{

max = den; e2 = e1;

v2 = v1;

}

}

}

//printf("%d\n", max); if(e2/v2 > 1) printf(">1\n"); else {

if(e2 == v2)

{

e2 = 1; v2 = 1;

}

cout << e2 << "/" << v2 << endl;

}

}

**DFS-Gudi trapped in the Room** #include<stdio.h> #include<string.h>

int res[1000000]; int Num[7];

int A,H,min;

void add(int \*a,int size )

{

int j,temp; for(j=2;j<=size;j++)

{

temp = a[j] + A; if(temp >= 10)

temp = temp % 10;

a[j] = temp; j++;

}

}

void Rotate(int \*a,int size)

{

int b[6],i,j,R; if(H > size) R = H - size; else

R = H;

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

{

b[i] = a[i];

}

for(i=R+1,j=1;i<=size;i++)

{

a[j] = b[i]; j++;

}

for(i=1;i<=R;i++,j++)

{

a[j] = b[i];

}

}

int getnum(int \*a,int size)

{

int temp=0,i; for(i=1;i<=size;i++)

{

temp += a[i]; temp \*= 10;

}

return (temp/10);

}

void copy(int \*a,int \*b,int size)

{

int i;

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

{

b[i] = a[i];

}

}

void lexico(int \*a,int size)

{

int Na = getnum(a,size); int cp[7];

if( res[Na] == 1) return;

res[Na] = 1;

if(Na < min) min = Na;

if(Na == 0) return;

if(A != 10)

{

copy(a,cp,size);

add(cp,size); lexico(cp,size);

}

copy(a,cp,size); Rotate(cp,size); lexico(cp,size);

}

int main()

{

char s[6],p[7]; int len,i,l,val,T; scanf("%d",&T); for(l=0;l<T;l++)

{

scanf("%s",s);

scanf("%d %d",&A,&H); len = strlen(s);

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

{

Num[i+1] = s[i] - 48;

}

val = getnum(Num,len); min = val;

lexico(Num,len); for(i=len-1;i>=0;i--)

{

p[i] = (min%10)+48;

min = min/10;

}

p[len] = '\0';

printf("%s\n",p); memset(res,0,1000000); memset(Num,0,7);

}

return 0;

}

**DFS-Feasible relations** #include<bits/stdc++.h> using namespace std; int tests;

string st[1<<20]; int a[1<<20];

int n,m,w[1<<20]; int b[1<<20];

int er;

int get(int x)

{

if (x==w[x])

return x; return w[x]=get(w[x]);

}

void merge(int a,int b)

{

a=get(a);

b=get(b);

w[a]=b;

}

int main(){ ios\_base::sync\_with\_stdio(0); cin>>tests;

for (;tests;--tests)

{

cin>>n>>m;

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

w[i]=i;

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

{

}

er=0;

cin>>a[i]>>st[i]>>b[i];

if (st[i]=="=")

merge(a[i],b[i]);

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

}

return 0;

}

{

}

if (er) else

if (st[i]=="=")

continue;

if (get(a[i])==get(b[i])) er=1;

cout<<"NO"<<endl; cout<<"YES"<<endl;

#### Minimum Spanning Tree-Friendless Dr. Sheldon Cooper

#include <cstdio> using namespace std;

int main()

{

int T; scanf("%d", &T);

for (int t = 0; t < T; ++t) { int A, B; scanf("%d%d",&A,&B); for (int i = 0; i < B; ++i) {

int m, n;

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

//printf("%d %d\n", m, n);

}

printf("%d\n", A-1);

}

return 0;

}

#### Connected or not-DFS

#include<stdio.h>

int a[20][20],reach[20],n; void dfs(int v){

int i; reach[v]=1; for(i=1;i<=n;i++)

if(a[v][i]&&!reach[i]){

printf("%d->%d\n",v,i); dfs(i);

}

}

int main(){

int i,j,count=0;

//printf(“\nEnter no of vertices : “);

scanf("%d",&n); for(i=1;i<=n;i++)

for(j=1;j<=n;j++){ reach[i]=0; a[i][j]=0;

}

//printf(“\nEnter adjacency matrix : \n”); for(i=1;i<=n;i++)

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

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

if(reach[i]) count++; if(count==n)

printf("Graph is connected"); else

printf("Graph is disconnected");

return 0;

}

#### Minimum Spanning Tree-Mr.President

#include <stdbool.h> #include <stdio.h> #include <stdlib.h>

#define MAX\_M (1000000)

#define MAX\_N (1000000) typedef unsigned long long ull; typedef struct { int x, y, c; } edge;

edge es[MAX\_M]; int p[MAX\_N+1]; int tk[MAX\_N+1]; int ccs,i;

int cmp(const void \* arg1, const void \* arg2) {

return ((const edge \*)arg1)->c - ((const edge \*)arg2)->c;

}

void dsu\_init(int n) { for (i=1; i <= n; ++i ) p[i] = i;

}

int dsu\_set(int x) { if ( p[x] == x ) return x;

return p[x] = dsu\_set(p[x]);

}

void dsu\_union(int x, int y) { int px = dsu\_set(x);

int py = dsu\_set(y);

if ( px == py ) return;

p[px] = py; ccs -= 1;

}

int main() { ull k,sum;

int i,n,m,a,b,c,acc;

scanf("%d %d %llu", &n, &m, &k); if ( n-1 > k ) {

printf("-1\n");

return EXIT\_SUCCESS;

}

dsu\_init(n);

for ( i=0, ccs=n; i < m; ++i ) { scanf("%d %d %d", &a, &b, &c); es[i] = (edge) { .x = a, .y = b, .c = c }; dsu\_union(a,b);

}

if ( ccs > 1 ) { printf("-1\n");

return EXIT\_SUCCESS;

}

dsu\_init(n);

qsort(es, m, sizeof(\*es), cmp); for ( i=sum=0; i < m; ++i )

if ( dsu\_set(es[i].x) != dsu\_set(es[i].y) )

dsu\_union(es[i].x, es[i].y), tk[i] = true, sum += es[i].c; for ( i=m-1, acc=0; i >= 0 && sum > k; --i )

if ( tk[i] )

sum -= (es[i].c-1), ++acc;

printf("%d\n", acc); return EXIT\_SUCCESS;

}

#### Minimum Spanning Tree-Pilgrims and Portals

#include<stdio.h>

#define INF 100000000000000

int x[200],done[200];

long long dist[200][200],graph[200][200]; void floyd (int n)

{

int i, j, k;

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

dist[i][j] = graph[i][j]; for (k = 0; k < n; k++)

{

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

{

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

{

if (dist[i][k] + dist[k][j] < dist[i][j])

dist[i][j] = dist[i][k] + dist[k][j];

}

}

}

}

int main()

{

int n,m,a,b,i,j; FILE \*p=stdin; int t;

long long c; fscanf(p,"%d",&t); while(t--)

{

int z=0,k;

long long ans=0; fscanf(p,"%d%d%d",&n,&m,&k); for(i=0; i<n; i++)

for(j=0; j<n; j++) graph[i][j]=0;

while(m--)

{

fscanf(p,"%d %d %lld",&a,&b,&c); graph[a-1][b-1]=c;

graph[b-1][a-1]=c;

}

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

{

if(graph[i][j]==0&&i!=j) graph[i][j]=INF;

}

floyd(n);

x[0]=1;

x[1]=0; dist[0][0]=INF;

for(i=0;i<n;i++) done[i]=0;

done[0]=1; for(i=1;i<k;i++)

{

int ans1=0,ans2=0; for(j=1;j<=x[0];j++)

{

for(z=1;z<k;z++) if(!done[z]&&dist[x[j]][z]<dist[ans1][ans2])

ans1=x[j],ans2=z;

}

x[++x[0]]=ans2; done[ans2]=1; ans+=dist[ans1][ans2];

}

printf("%lld\n",ans);

}

return 0;

}

#### Minimum Spanning Tree-Quantitative coefficient

#include <stdio.h> #include <stdlib.h>

typedef unsigned long long ull; #define QUOTIENT (1000000007)

#define MAX\_M (31313)

#define MAX\_N (1500)

typedef struct { int u, v, c; } edge;

edge es[MAX\_M]; int p[MAX\_N+1]; int ccs;

int cmp(const void \* arg1, const void \* arg2) {

return ((const edge \*)arg1)->c - ((const edge \*)arg2)->c;

}

void dsu\_init(int n) { int i;

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

}

int dsu\_set(int x) {

if ( p[x] == x ) return x; return p[x] = dsu\_set(p[x]);

}

void dsu\_union(int x, int y) { int px = dsu\_set(x);

int py = dsu\_set(y);

if ( px == py ) return; p[px] = py;

ccs -= 1;

}

int main() { ull acc;

int i,t,n,m;

scanf("%d", &t);

while ( t-- ) {

scanf("%d %d", &n, &m); dsu\_init(n);

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

scanf("%d %d %d", &es[i].u, &es[i].v, &es[i].c); qsort(es, m, sizeof(\*es), cmp);

for ( i=0, acc=1, ccs=n; ccs > 1; ++i )

if ( dsu\_set(es[i].u) != dsu\_set(es[i].v) ) { acc = (acc \* es[i].c) % QUOTIENT; dsu\_union(es[i].u, es[i].v);

}

printf("%llu\n", acc);

}

return EXIT\_SUCCESS;

}

#### Hamiltonian Path-Micro and Coins

#include<stdio.h> int n,m;

int x[1000000],y[1000000],w[1000000],dp[1000000][100];

int getMin(int a,int b){

if(b==-1)

if(a==-1)

return a;

return b;

return a<b?a:b;

}

int getWeight(int a, int b){

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

if(x[i]==a && y[i]==b){

return 1;

}else if(x[i]==b && y[i]==a){ return 1;

}

}

return 0;

}

int main(void){

int t; scanf("%d",&t);

while(t--){

int i,j,mask; scanf("%d%d",&n,&m); for(int i=0;i<m;i++)

scanf("%d%d",&x[i],&y[i]); int end=(1<<n);

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

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

dp[i][j]=0;

}

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

dp[1<<i][i]=1;

}

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

dp[0][i]=1;

for(mask=1;mask<end;mask++){ for(i=0;i<n;i++){

if(!(mask&(1<<i))) continue; int temp=mask-(1<<i); for(j=0;j<n;j++){

if(!(temp&(1<<j))) continue;

int weight=getWeight((i+1),(j+1)); if(weight!=0 && dp[temp][j]!=0){ dp[mask][i]=1;

}

}

}

}

int flag=1; for(i=0;i<end;i++){

if(dp[end-1][i]==1){

flag=0; break;

}

}

if(flag==0)

printf("Yes\n");

}

return 0;

}

else

printf("No\n");

#### Hamiltonian Path-Micro and Permutations

#include<stdio.h> int n,m;

int x[1000000],y[1000000],w[1000000],dp[1000000][100];

int getMin(int a,int b){

if(b==-1)

if(a==-1)

return a;

return b;

return a<b?a:b;

}

int getWeight(int a, int b){

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

if(x[i]==a && y[i]==b){

return 1;

}else if(x[i]==b && y[i]==a){ return 1;

}

}

return 0;

}

int main(void){

int i,j,mask; scanf("%d%d",&n,&m);

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

scanf("%d%d",&x[i],&y[i]); int end=(1<<n);

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

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

dp[i][j]=0;

}

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

dp[1<<i][i]=1;

}

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

dp[0][i]=1;

for(mask=1;mask<end;mask++){ for(i=0;i<n;i++){

if(!(mask&(1<<i))) continue; int temp=mask-(1<<i); for(j=0;j<n;j++){

if(!(temp&(1<<j))) continue;

int weight=getWeight((i+1),(j+1)); if(weight!=0 && dp[temp][j]!=0){

dp[mask][i]+=dp[temp][j];

}

}

}

}

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

count+=dp[end-1][i];

}

printf("%d",count); return 0;

}

**DFS-Jungle Run** #include <stdio.h> #include <limits.h> #define UP 1

#define DOWN 2

#define LEFT 3

#define RIGHT 4 int mat[30][30];

int min=INT\_MAX,n; char arr[30][30];

void minPath(int i,int j,int status,int k)

{

if(arr[i][j]=='T')

{

return;

}

if(arr[i][j]=='E')

{

if(k<min) min=k; else return;

}

if(mat[i][j])

{

if(k<mat[i][j])

{

mat[i][j]=k;

}

else return;

}

else mat[i][j]=k;

if(i-1>=0 && status!=DOWN)

{

minPath(i-1,j,UP,k+1);

}

if(i+1<n && status!=UP)

{

minPath(i+1,j,DOWN,k+1);

}

if(j-1>=0 && status!=RIGHT)

{

minPath(i,j-1,LEFT,k+1);

}

if(j+1<n && status!=LEFT)

{

minPath(i,j+1,RIGHT,k+1);

}

}

int main(void) { int i,j,iE,jE; char c;

scanf("%d",&n); c=getchar(); for(i=0;i<n;i++)

{

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

{

arr[i][j]=getchar(); c=getchar(); if(arr[i][j]=='S')

{

mat[i][j]=0; iE=i;

jE=j;

}

}

}

if(iE-1>=0)

{

minPath(iE-1,jE,UP,1);

}

if(iE+1<n )

{

minPath(iE+1,jE,DOWN,1);

}

if(jE-1>=0)

{

minPath(iE,jE-1,LEFT,1);

}

if(jE+1<n )

{

minPath(iE,jE+1,RIGHT,1);

}

printf("%d",min); return 0;

}

#### Hamiltonian Path-Fredo and his Birthday Gift

#include "bits/stdc++.h" using namespace std;

const int maxn=17; int n;

int arr[maxn][maxn];

int dp[1<<maxn][maxn];

int HamiltonianCycle(int mask,int pre){ int ans=0;

if(dp[mask][pre]!=-1)return dp[mask][pre]; for(int j=1;j<=n;j++){

if(!(mask&(1<<j))&&arr[pre][j]){

int ret=1+HamiltonianCycle(mask|(1<<j),j);

ans=max(ret,ans);

}

}

return dp[mask][pre]=ans;

}

int main(){

//freopen("t.txt","r",stdin); int m,u,v,t; for(scanf("%d",&t);t--;){

scanf("%d%d",&n,&m); memset(arr,0,sizeof arr); memset(dp,-1,sizeof dp); for(int i=1;i<=n;i++)arr[i][i]=1; for(int i=0;i<m;i++){

scanf("%d%d",&u,&v);

arr[u][v]=1;

arr[v][u]=1;

}

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

printf("%d ",HamiltonianCycle(1<<i,i));

}

puts("");

}

}

**BFS-Big P and Party** #include<bits/stdc++.h> using namespace std; int a;

long int b; bool vis[1002];

int lucky[1002]; int main()

{cin>>a>>b;

for(int i=0;i<1002;i++) lucky[i]=-1; vector<int>g[1002]; int u,v;

for(int i=0;i<b;i++)

{

cin>>u>>v; g[u].push\_back(v);

g[v].push\_back(u);

}

lucky[0]=0;

for(int i=0;i<1002;i++)

vis[i]=false; queue<int>q; q.push(0); vis[0]=true; while(!q.empty())

{

int w=q.front(); q.pop();

for(int i=0;i<g[w].size();i++)

{ if(!vis[g[w][i]])

{ q.push(g[w][i]);

vis[g[w][i]]=true;

lucky[g[w][i]]=lucky[w]+1;

}

}

}

for(int i=1;i<a;i++) cout<<lucky[i]<<endl;

return 0;

}

**BFS-We Are On Fire** #include<bits/stdc++.h> using namespace std; int ar[1001][1001];

bool vis[1001][1001]; int n, m;

int dx[] = {-1,1,0,0};

int dy[] = {0,0,1,-1};

int dfs(int i , int j )

{

if(i<1|| i>n || j<1 || j>m) return 0;

if(vis[i][j]) return 0;

if(ar[i][j]==0) return 0;

int cnt = 1; vis[i][j] = true;

for(int k=0;k<4;k++)

cnt+=dfs(i+dx[k],j+dy[k]); return cnt;

}

int main()

{

ios::sync\_with\_stdio(false); cin.tie(0);

int q,x,y; int cnt = 0;

cin>>n>>m>>q;

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

{

cin>>ar[i][j];

if(ar[i][j]==1) cnt++;

vis[i][j] = false;

}

for(int i=1;i<=q;i++)

{

cin>>x>>y;

if(ar[x][y]==0 || vis[x][y]) goto p;

cnt-=dfs(x,y); p:

cout<<cnt<<'\n';

}

}

**BFS-Permutation Swap** #include <stdio.h> #include <stdbool.h>

#define MAX\_N 101000 int P[MAX\_N], Q[MAX\_N];

int Id[MAX\_N], sz[MAX\_N];

int find( int x ) {

if( x != Id[x] ) Id[x] = find(Id[x]); return Id[x];

}

void Union( int x, int y ) {

x = find(x), y = find(y); if( x == y ) return;

if( sz[x] >= sz[y] ) {

Id[y] = x; sz[x] += sz[y];

}

else {

}

}

Id[x] = y; sz[y] += sz[x];

int main( void ) {

int T; scanf("%i", &T);

while( T-- ) {

int n, m, i;

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

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

scanf("%i", &P[i]);

for( i = 1; i <= n; i++ ) { int q; scanf("%i", &q); Q[q] = i;

}

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

Id[i] = i, sz[i] = 1;

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

int u, v;

scanf("%i %i", &u, &v); Union(u, v);

}

bool isPossible = true; for( i = 1; i <= n; i++ ) {

int p = P[i];

if( find(i) != find(Q[p]) )

isPossible = false;

}

puts( isPossible ? "YES" : "NO" );

}

return 0;

}

**Journey to the Moon** #include <iostream> #include <vector>

static inline int Count(const std::vector<int> &groups, int head)

{

return -groups[head];

}

static int Find(std::vector<int> &groups, int node)

{

if (groups[node] < 0) return node;

return (groups[node] = Find(groups, groups[node]));

}

static void Union(std::vector<int> &groups, int a, int b)

{

int parentA = Find(groups, a); int parentB = Find(groups, b);

if (parentA == parentB) return;

int countA = Count(groups, parentA); int countB = Count(groups, parentB); int newCount = countA + countB;

if (countA > countB) { groups[parentB] = parentA; groups[parentA] = -newCount;

} else {

groups[parentA] = parentB; groups[parentB] = -newCount;

}

}

int main(void)

{

int n, p;

std::cin >> n >> p; std::vector<int> groups(n, -1);

for (int i = 0; i < p; i++) { int a, b;

std::cin >> a >> b; Union(groups, a, b);

}

long long total = 0; long long prevCount = 0; for (int i = 0; i < n; i++) {

if (Find(groups, i) == i) {

int count = Count(groups, i);

total += count \* prevCount; prevCount += count;

}

}

std::cout << total << "\n";

}

#### Even Tree

#include <stdio.h>

long long a[200][200],b[200][2],i,j,k,l,m,n; int main()

{

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

for(i=1;i<=n;i++) b[i][1]=1; while(m--)

{

scanf("%lld %lld",&i,&j); a[i][j]= a[j][i] =1; b[i][0]++;

b[j][0]++;

} m=0;

while(1)

{ k=1;

for(i=1;i<=n;i++) if(b[i][0] == 0)

{ k=0;

if(b[i][1]%2==1) m=-10000;

b[i][0]=-1;

}

for(i=1;i<=n;i++) if(b[i][0] == 1)

{ k=0; j=1;

while(a[i][j]==0) j++;

if(b[i][1]%2==0)

{ m++;

// printf("%lld %lld..\n",i,j);

a[i][j] = a[j][i] = 0;

b[j][0]--;

b[i][0]=-1;

}

else

{

a[i][j] = a[j][i] = 0;

b[j][0]--;

b[j][1] += b[i][1]; b[i][0]=-1;

}

}

if(k) break;

}

if(m<0) printf("-1\n"); else printf("%lld\n",m); return 0;

}

#### Snakes and Ladders: The Quickest Way Up

#include<bits/stdc++.h> using namespace std;

int n, m; queue<int> q;

int go\_immediately\_to[110], dist[110]; bool vis[110];

bool isValid(int node)

{

if(node < 1 || node > 100 || vis[node]) return false;

else

return true;

}

int BFS(int source)

{

memset(vis, 0, sizeof(vis)); while(!q.empty())

q.pop();

vis[source] = 1; q.push(source); dist[source] = 0; while(!q.empty())

{

int current\_node = q.front(); q.pop();

for(int i = 1; i<=6; i++)

{

int next\_node = go\_immediately\_to[current\_node+i]; if(isValid(next\_node))

{

q.push(next\_node); vis[next\_node] = 1;

dist[next\_node] = dist[current\_node]+1;

}

}

}

if(!vis[100]) return -1;

else

return dist[100];

}

int main()

{

int i, j, cs, t, u, v; cin >> t;

for(cs = 1; cs<=t; cs++)

{

cin >> n;

for(i = 1; i<=100; i++) go\_immediately\_to[i] = i;

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

{

cin >> u >> v; go\_immediately\_to[u] = v;

}

cin >> m;

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

{

cin >> u >> v; go\_immediately\_to[u] = v;

}

cout << BFS(1) << endl;

}

}

#### Prim's (MST) : Special Subtree

#include <stdio.h> #include <string.h> #include <math.h> #include <stdlib.h>

int main() {

int i,j,k,le,b[3000],n,m,x,y,s,r,no,sum=0; int \*a[3000];

for(i=0;i<=3000;i++) a[i]=(int\*)malloc(3000\*(sizeof(int)));

scanf("%d%d",&n,&m); for(i=1;i<=n;i++){

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

{ a[i][j]=100001;

}b[i]=0;}

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

{

scanf("%d%d%d",&x,&y,&r); a[x][y]=r;

a[y][x]=r;

}

scanf("%d",&s); b[s]=-1;

for(k=1;k<n;k++){le=100001; for(i=1;i<=n;i++)

{

if(b[i]==-1)

{

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

{

if((a[i][j]<le)&&(b[j]==0)){ le=a[i][j];no=j;}

}

}

}

sum=sum+le;

// printf("%d\n",le); b[no]=-1;

}

printf("%d",sum);

/\* Enter your code here. Read input from STDIN. Print output to STDOUT \*/ return 0;

}

**Jack goes to Rapture** #include<iostream> #include<bits/stdc++.h>

using namespace std;

vector<int> parent(50010); vector<int> rank1(50010); vector<pair<int,int> > Edge[50005]; int visited[50005];

int global\_var = 0;

void dfs(int x , int value , int target)

{

if(visited[x] ==1 ) return; if(x==target)

{

cout<<(int)value<<endl;

//global\_var = 1; exit(0);

}

visited[x] = 1;

vector<pair<int,int > > :: iterator it;

for(it = Edge[x].begin() ; it<Edge[x].end() ; it++ )

{

if((\*it).second > value)

{

dfs((\*it).first ,(\*it).second , target);

}

else

{

dfs((\*it).first ,value , target);

}

}

}

void initSet(int \_size){ int i; parent.resize(\_size); rank1.resize(\_size);

for(i=0;i<\_size;i++){ parent[i]=i; rank1[i]=0;

}

}

int findset(int i){ if(parent[i] !=i){

parent[i]=findset(parent[i]);

}

return parent[i];

}

int union\_(int u,int v){ int r1,r2; r1=findset(u); r2=findset(v);

if(rank1[r1]>rank1[r2]){ parent[r2]=r1;

}

else if(rank1[r1]<rank1[r2]){ parent[r1]=r2;

}

else{ parent[r2]=r1;

rank1[r1]++;

}

}

bool isSameSet(int u,int v){ return findset(u)==findset(v);

}

class Compare\_priority{ public:

bool operator()(const pair<float,pair<int,int> > &p1,const pair<float,pair<int,int> > &p2){ if(p1.first>p2.first) return true;

else

return false;

}

};

int main()

{

int N,edges,i,p,q; float weight; cin>>N; initSet(N);

priority\_queue< pair<float,pair<int,int> >, vector< pair<float,pair<int,int> > > ,Compare\_priority > Edges; cin>>edges;

for(i=0;i<edges;i++){ cin>>p>>q>>weight;

Edges.push(make\_pair(weight,make\_pair(p,q)));

}

for(i=1 ; i<=N ; i++) visited[i] = 0;

while(!Edges.empty()){

pair<float,pair<int,int> > P=Edges.top();Edges.pop();

if(!isSameSet(P.second.first,P.second.second)){ union\_(P.second.first,P.second.second); Edge[P.second.first].push\_back(make\_pair(P.second.second, P.first)); Edge[P.second.second].push\_back(make\_pair(P.second.first,P.first));

// cout<<"("<<P.second.first<<","<<P.second.second<<")"<<endl;

// cost +=P.first;

}

}

dfs(1,-100000,N);

cout<<"NO PATH EXIST"<<endl;

}

**BFS-Utkarsh in Gardens** #include <bits/stdc++.h> using namespace std; const int MAXN = 2018;

bitset<MAXN> g[MAXN], com; int n;

int main()

{

scanf("%d", &n);

assert(1 <= n && n <= 2000); for (int i = 1; i <= n; i++) {

for (int j = 1; j <= n; j++) { int x; scanf("%d", &x);

//assert(x == 0 || x == 1); g[i][j] = x;

}

}

long long ans = 0;

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

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

long long cnt = 0;

//cout<<"g[i] "<<i<<" "<<g[i]<<" g[j]"<<j<<" "<<g[j]<<endl; cnt = (g[i] & g[j]).count();

//cout<<"cnt "<<cnt<<endl;

ans += cnt\*(cnt - 1) / 2;

}

}

cout<<ans/2<<endl; return 0;

}

#### BFS-Sonya and the graph with disappearing edges

#include <bits/stdc++.h>

using namespace std; int inf = 100000000;

int visited[100010], timee[100010]; vector<int> v[100001]; map<pair<int, int>, int> mp;

int main()

{

int n, m, tq;

cin >> n >> m >> tq; for(int i=0; i<=n; i++)

{

visited[i] = 0; timee[i] = INT\_MAX;

}

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

{

int a, b;

cin >> a >> b; mp[make\_pair(a, b)] = i; mp[make\_pair(b, a)] = i; v[a].push\_back(b);

v[b].push\_back(a); timee[i] = inf;

}

while(tq--)

{

int a, b;

cin >> a >> b; timee[b] = a;

}

queue<pair<int, int> > q; q.push(make\_pair(1, 0));

visited[1] = 1; int ans = -1;

while(!q.empty())

{

int a = q.front().first;

int b = q.front().second; q.pop();

if(a==n)

{

ans = b; break;

}

for(int i=0; i<v[a].size(); i++)

{

if(visited[v[a][i]]==0 && timee[mp[make\_pair(a, v[a][i])]]>=(b+1))

{

q.push(make\_pair(v[a][i], b+1)); visited[v[a][i]] = 1;

}

}

}

cout << ans << endl;

//cout << "Hello world!" << endl;

return 0;

}

#### BFS-The Witches of Hogwarts!

#include <bits/stdc++.h> using namespace std; #define LL long long

int main()

{

int t; cin>>t; while(t--)

{

LL n;

cin>>n; queue <LL> q;

map <LL,int> mp; q.push(n);

mp[n]=0; while(!q.empty())

{

LL tmp=q.front(); q.pop(); if(tmp==1)

{

cout<<mp[tmp]<<endl; break;

}

if(tmp>1)

{

if(mp.find(tmp-1)==mp.end())

{

mp[tmp-1]=mp[tmp]+1; q.push(tmp-1);

}

if(tmp%2==0 && mp.find(tmp/2)==mp.end())

{

mp[tmp/2]=mp[tmp]+1; q.push(tmp/2);

}

if(tmp%3==0 && mp.find(tmp/3)==mp.end())

{

mp[tmp/3]=mp[tmp]+1; q.push(tmp/3);

}

}

}

}

return 0;

}

# DAA E-Lab

**Session-8**

**Xsquare And Two Strings** #include<stdio.h> #include<string.h>

int main(){

int tc,found,inner,len;

char str1[100005],str2[100005],strset[27]; scanf("%d",&tc);

while(tc-- >0){ found=0;

for(inner=0;inner<27;inner++){ strset[inner]=0;

}

scanf("%s",str1);

scanf("%s",str2); len=strlen(str1); for(inner=0;inner<len;inner++){

strset[str1[inner]-97]=strset[str1[inner]-97]+1;

}

len=strlen(str2); for(inner=0;inner<len;inner++){

if(strset[str2[inner]-97] != 0){ found++;

strset[str2[inner]-97]=strset[str2[inner]-97]-1;

}

}

if(found>=2)

printf("Yes\n");

else

}

return 0;

}

printf("No\n");

#### Subset of the word

#include<stdio.h> char string[50]; int n;

void subset(int,int,int);

int main()

{

int i,len;

// printf("Enter the len of main set : "); scanf("%d",&len);

// printf("Enter the elements of main set : "); scanf("%s",string);

n=len;

// printf("The subsets are :\n"); for (i=1;i<=n;i++)

subset(0,0,i); return 0;

}

/\*Function to find the number of subsets in the given string\*/ void subset(int start,int index,int num\_sub)

{

int i,j;

if(index-start+1==num\_sub)

{

if(num\_sub==1)

{

for(i=0;i<n;i++) printf("%c\n",string[i]);

}

else

{

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

{

for (i=start;i<index;i++) printf("%c",string[i]);

printf("%c\n",string[j]);

}

if (start!=n-num\_sub)

subset(start + 1,start + 1,num\_sub);

}

}

else

{

subset(start,index + 1,num\_sub);

}

}

**Little Monk and Good String** #include<iostream> #include<algorithm>

using namespace std;

int isVowel(char c)

{

if(c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u')

{

return 1;

}

return 0;

}

int findSubstring(string s)

{

int res = 0, tc = 0, len = s.size();

for(int i=0; i<len; i++)

{

if(isVowel(s[i]))

{

tc++;

if(tc > res)

{

res = tc;

}

}

else

{

tc = 0;

}

}

return res;

}

int main()

{

string s; cin>>s;

cout<<findSubstring(s)<<"\n"; return 0;

}

**Matching Strings** #include <iostream> using namespace std;

int main()

{

int t; cin >>t;

while(t--)

{

int count=0,i=0; string s,r;

cin >>s>>r;

int ls=s.length(); int lr=r.length(); while(i<ls && i<lr)

{

if(s[i]==r[i]) count++; i++;

}

cout<<count<<endl;

}

return 0;

}

**Naive Pattern Searching** #include<stdio.h> #include<string.h>

void search(char pat[30],char txt[30])

{

int M=strlen(pat); int i,j;

int N=strlen(txt); for( i=0;i<N-M+1;i++)

{

for(j=0;j<M;j++) if(txt[i+j]!=pat[j]) break;

if(j==M)

printf("Pattern found at index %d \n",i);

}

}

int main()

{

char txt[30],pat[30]; fgets(txt,30,stdin); fgets(pat,30,stdin);

search(pat,txt);

return 0;

}

**Modified Naive Pattern Searching** #include <stdio.h> #include<string.h>

#define MAX 100 int main()

{

char str[50],s[50];

int i,j,k,l,sl,eq;

scanf("%[^\n]%\*c",str);

//printf("Text: %s\n",str);

//fgets(s, MAX, stdin); scanf("%[^\n]%\*c",s);

// printf("Pattern: %s\n",s); l=strlen(str);

sl=strlen(s); if(strcmp(str,s)==0)

{

printf("Pattern starts at posn 1");

}

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

{

eq=0; if(str[i]!=s[0])

{

continue;

}

else

{

for(j=i,k=0;j<i+sl,k<sl;j++,k++)

{

if(str[j]!=s[k])

{

eq=-1; break;

}

}

}

if (eq==0 && k>=sl)

{

printf("Pattern found at index %d\n",i);

}

}

return 0;

}

#### Rabin-Karp Algorithm fpr String Searching

#include <iostream> #include<cstring> using namespace std;

void search(char pat[],char txt[])

{

int m=strlen(pat); int n=strlen(txt);

for(int i=0;i<=n-m;i++)

{

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

{

if(pat[j]!=txt[i+j]) break;

if(j==m-1)

cout<<"Pattern found at index "<<i<<endl;

}

}

}

int main()

{

char pat[50],txt[50]; cin.getline(txt,20); cin>>pat;

search(pat,txt);

}

#### Pattern Matching

#include <stdio.h>

int match(char\*, char\*);

int main()

{

char a[100], b[100];

int position; scanf("%[^\n]s", a);

// puts(a); scanf("%s",b);

position = match(a, b); if(position!=-1)

printf("Found at location %d\n", position+1);

else

printf("Not found.\n");

return 0;

}

int match(char \*a, char \*b)

{

int c;

int position = 0; char \*x, \*y;

x = a; y = b;

while(\*a)

{

while(\*x==\*y)

{

x++; y++;

if(\*x=='\0'||\*y=='\0') break;

}

if(\*y=='\0') break;

a++;

position++; x = a;

y = b;

}

if(\*a)

return position; else

return -1;

}

**Brute-Force Pattern Matching** #include <stdio.h> #include<string.h>

#define MAX 100 int main()

{

char str[50],s[50];

int i,j,k,l,sl,eq;

scanf("%[^\n]%\*c",str); printf("Text: %s\n",str);

//fgets(s, MAX, stdin); scanf("%[^\n]%\*c",s); printf("Pattern: %s\n",s); l=strlen(str);

sl=strlen(s); if(strcmp(str,s)==0)

{

printf("Pattern starts at posn 1");

}

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

{

eq=0; if(str[i]!=s[0])

{

continue;

}

else

{

for(j=i,k=0;j<i+sl,k<sl;j++,k++)

{

if(str[j]!=s[k])

{

eq=-1; break;

}

}

}

if (eq==0 && k>=sl)

{

printf("Pattern starts at posn %d",i+1); break;

}

}

return 0;

}

#### Brute-Force String Matching Problem

#include <iostream> #include <string.h> using namespace std;

int match(char [], char []); int main() {

char a[100], b[100]; int position; cin.getline(a,100); cin.getline(b,100); position = match(a, b);

if(position != -1)

{

cout<<"Text: "<<a<<"\n"; cout<<"Pattern: "<<b<<"\n";

cout<<"Pattern starts at posn "<< position + 1;

}

/\* else {

printf("Not found.\n");

}

\*/

return 0;

}

int match(char text[], char pattern[]) {

int c, d, e, text\_length, pattern\_length, position = -1;

text\_length = strlen(text); pattern\_length = strlen(pattern);

if (pattern\_length > text\_length) { return -1;

}

for (c = 0; c <= text\_length - pattern\_length; c++) { position = e = c;

for (d = 0; d < pattern\_length; d++) { if (pattern[d] == text[e]) {

e++;

}

else { break;

}

}

if (d == pattern\_length) { return position;

}

}

return -1;

}

#### Knuth-Morris-Pratt algorithm for Pattern Matching

**(CAN’T FIND THE CODE)**

**String Matching Using String Library**

#include <stdio.h> #include <string.h>

int main()

{

char str1[5], str2[5]; int result,i,j; scanf("%s",str1);

scanf("%s",str2);

// comparing strings str1 and str2 result = strcmp(str1, str2);

// printf("strcmp(str1, str2) = %d\n", result);

if(result!=0){

printf("Entered strings are not equal");

}else{

printf("Entered strings are equal");

}

return 0;

}

#### Compare

#include <stdio.h> #include<string.h> int main()

{

char a[30],b[30];

scanf("%s%s",a,b); if(strcmp(a,b)==0)

printf("Entered strings are equal."); else

printf("Entered strings are not equal."); return 0;

}

#### String Occurrence

#include <stdio.h>

#include <string.h> int main()

{

int i,j,m,n,count=0,count1=0; char str[100],sub[100]; scanf("%[^\n]s",sub); m=strlen(sub); scanf("%s",str);

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

{

j=0;

count=0; while((sub[j]==str[i])&&(sub[j]!='\0'))

{

count++; i++;

j++;

}

if(count==m)

{

count1++; count=0;

}

else

i++;

}

printf("\n%d",count1); return 0;

}

**Playful String** #include <stdio.h> #include <string.h> #include <stdbool.h>

char str[100000],sub[100000]; bool isSubstring()

{

int i=0,j=0;

while(str[i]!='\0' && sub[j]!='\0'){ if(str[i]==sub[j]){

i++; j++;

}

else

i++;

}

if(j==strlen(sub)) return true;

return false;

}

int main()

{

int t,i,j,k; char ch;

scanf("%d",&t); for(i=1;i<=t;i++){

scanf("%s",str);

scanf("%s",sub);

if(isSubstring()){ j=0;

k=strlen(sub)-1; while(j<k){

char ch=sub[j]; sub[j]=sub[k]; sub[k]=ch;

j++; k--;

}

if(isSubstring()) printf("GOOD STRING");

else

printf("BAD STRING");

}

else

printf("BAD STRING"); printf("\n");

}

return 0;

}

#### HackerRank in a String!

#include <bits/stdc++.h>

using namespace std; int main(){

int q; cin >> q;

for(int a0 = 0; a0 < q; a0++){ string s;

cin >> s;

string cur = "hackerrank"; int st = 0;

for (int i= 0; i < s.size() && st < cur.size(); i++) { if (s[i] == cur[st]) {

st++;

}

}

if (st == cur.size()) {

cout << "YES" << endl;

} else {

cout << "NO" << endl;

}

// your code goes here

}

return 0;

}

**Pangrams** #include<stdio.h> char st[100000]; int i,ind[1000]; int main()

{

while(fgets(st,100,stdin))

{

for(i='A';i<='Z';i++)

ind[i]=0;

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

{

if(st[i]>='a' && st[i]<='z') st[i]-=32;

ind[st[i]]++;

}

for(i='A';i<='Z';i++)

if(ind[i]==0) break;

if(i=='Z'+1)

printf("pangram\n"); else

printf("not pangram\n");

}

return 0;

}

**Gems** #include<bits/stdc++.h> using namespace std; int main()

{

int n,i,ans=0,ar[109][26]={},j,flag; cin >> n;

string s;

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

{

cin >> s;

for(j=0; j<s.size(); j++) ar[i][s[j]-'a']++;

}

for(i=0; i<26; i++)

{

flag=0;

for(j=0; j<n; j++) if(ar[j][i]==0)flag=1;

if(flag==0)ans++;

}

cout << ans << endl; return 0;

}

**Two Strings** #include <cmath> #include <cstdio> #include <vector> #include <iostream>

#include <algorithm> using namespace std;

int main() { int t; cin>>t; while(t--){

int flag=0; string s,s1; cin>>s>>s1;

for(int i=97;i<(97+25);i++)

{ char c=i;

if( s.find(c,0)!=std::string::npos && s1.find(c,0)!=std::string::npos)

{

flag = 1; break;

}

}

if(flag) cout<<"YES"; else cout<<"NO"; cout<<endl;

}

/\* Enter your code here. Read input from STDIN. Print output to STDOUT \*/ return 0;

}

#### PALINDROME COUNT

#include<bits/stdc++.h> using namespace std;

int CountPS(char str[], int n)

{

int dp[n][n];

memset(dp, 0, sizeof(dp));

bool P[n][n];

memset(P, false , sizeof(P));

for (int i= 0; i< n; i++) P[i][i] = true;

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

{

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

{

P[i][i+1] = true;

dp[i][i+1] = 1 ;

}

}

for (int gap=2 ; gap<n; gap++)

{

for (int i=0; i<n-gap; i++)

{

int j = gap + i;

if (str[i] == str[j] && P[i+1][j-1] ) P[i][j] = true;

if (P[i][j] == true)

dp[i][j] = dp[i][j-1] + dp[i+1][j] + 1 - dp[i+1][j-1]; else

dp[i][j] = dp[i][j-1] + dp[i+1][j] - dp[i+1][j-1];

}

}

return dp[0][n-1];

}

int main()

{

//char str[] = "abaab"; char str[20];

cin>>str;

//cout<<str;

int n = strlen(str);

cout << CountPS(str, n)+n << endl; return 0;

}

#### Joy And Two Strings

#include <iostream>

int main()

{

int t;

bool letters[26], common; std::string s1, s2;

std::cin >> t;

for (int i = 0; i < t; i++) {

std::cin >> s1 >> s2; common = false;

for (int j = 0; j < 26; j++) {

letters[j] = false;

}

for (int j = 0; j < s1.size(); j++) {

letters[s1[j] - 'a'] = true;

}

for (int j = 0; j < s2.size(); j++) { if (letters[s2[j] - 'a']) {

std::cout << "Yes" << std::endl; common = true;

break;

}

}

if (!common) {

std::cout << "No" << std::endl;

}

}

}

**Ram and Wow String** #include<stdio.h> #include<string.h> int main(){

char b[100005]; scanf("%s",b);

int j,l,count=0,ans=0; l=strlen(b); for(j=0;j<l;j++){

if(b[j]=='a'||b[j]=='e'||b[j]=='i'||b[j]=='o'||b[j]=='u'){ count++;

}

else{ count=0;} if(count>ans) ans=count;

}

printf("%d",ans); return 0;

}

**Mia and String Matching** #include <stdio.h> #include<string.h>

int main(void) { int t,i,a,b,c;

char str[10000],str2[10000]; scanf("%d",&t);

while(t--)

{

c=0;

scanf("%s",str);

scanf("%s",str2); a = strlen(str);

b = strlen(str2); if(a>=b)

{

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

{

if(str[i]==str2[i]) c++;

}

}

else if(a<b)

{

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

{

if(str[i]==str2[i]) c++;

}

}

printf("%d\n",c); for(i=0;i<b;i++) str2[i] = '\0';

}

return 0;

}

**Pattern Matching** #include <stdio.h> #include <string.h>

int match(char [], char []); int main() {

char a[100], b[100];

int position;

scanf("%[^\n]s", a);

scanf(" %[^\n]s", b); position = match(a, b);

if(position != -1) {

printf("Found at location %d\n", position + 1);

}

else {

printf("Not found.\n");

}

return 0;

}

int match(char text[], char pattern[]) {

int c, d, e, text\_length, pattern\_length, position = -1;

text\_length = strlen(text); pattern\_length = strlen(pattern);

if (pattern\_length > text\_length) { return -1;

}

for (c = 0; c <= text\_length - pattern\_length; c++) { position = e = c;

for (d = 0; d < pattern\_length; d++) { if (pattern[d] == text[e]) {

e++;

}

else { break;

}

}

if (d == pattern\_length) { return position;

}

}

return -1;

}

**strcmp for string matching** #include <stdio.h> #include <string.h>

int main()

{

char a[100], b[100];

scanf("%s",a);

scanf("%s",b);

if (strcmp(a,b) == 0)

printf("Entered strings are equal\n"); else

printf("Entered strings are not equal\n");

return 0;

}

**ELab in a String!** #include <stdio.h> #include<string.h> int main()

{

int t,i,j,cnt,l,l1;

char s[20],str[]="ELab";

scanf("%d",&t); while(t)

{

cnt=0; scanf("%s",s); l=strlen(s); l1=strlen(str); while(i<l1 && j<l)

{

if(str[i]==s[j])

{

i++;

cnt++;

}

else

{

j++;

}

}

if(cnt>=l1) printf("YES\n"); else printf("NO\n");

t--;

}

return 0;

}

**Ristha's Pangrams** #include <iostream> #include <string>

using namespace std; int main()

{

string inputString; getline(cin, inputString); int alphabetTab = 0;

int alphabet[26]; int flag = 0;

for(int i = 0; i < 26; i++){

alphabet[i] = 0;

}

for(int i = 0; i < inputString.length(); i++){

int insertNum = (tolower(inputString[i]) - 'a'); if(insertNum < 0 || insertNum > 25){

continue;

}

if(alphabet[insertNum] == 0){

alphabetTab++;

}

if(alphabetTab >= 26){

flag = 1; break;

} else {

}

}

alphabet[insertNum]++;

if(flag){

} else {

}

return 0;

}

cout << "pangram" << endl; cout << "not pangram" << endl;

#### Gemstones

#include<iostream> #include<string> using namespace std; int main()

{

int T;

int a[26] = {0};

bool flag[26] = {false}; int nCount = 0;

cin >> T; cin.ignore();

int curTest = 1; while(curTest <= T)

{

string in; getline(cin,in);

for(int i = 0; i < in.length();i++)

{

int ch = ((int)in[i]) - 97;

if( ch >= 0 && ch < 26 && flag[ch] == false)

{ a[ch]++;

flag[ch] = true;

}

}

for(int i = 0; i < 26;i++) flag[i] = false;

curTest++;

}

for(int i = 0 ; i <= 25;i++) if(a[i] == T)

nCount++; cout<<nCount;

return 0;

}

#### Naive-Recursive: Longest Common Subsequence

#include<iostream> #include<cstring> #include<cstdlib> using namespace std;

void lcs( char \*X, char \*Y, int m, int n )

{

int L[m+1][n+1];

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

{

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

{

if (i == 0 || j == 0) L[i][j] = 0;

else if (X[i-1] == Y[j-1])

L[i][j] = L[i-1][j-1] + 1;

else

L[i][j] = max(L[i-1][j], L[i][j-1]);

}

}

int index = L[m][n];

char lcs[index+1]; lcs[index] = '\0';

int i = m, j = n;

while (i > 0 && j > 0)

{

if (X[i-1] == Y[j-1])

{

lcs[index-1] = X[i-1];

i--; j--; index--;

}

else if (L[i-1][j] > L[i][j-1]) i--;

else j--;

}

cout << "Length of LCS is " << strlen(lcs);

}

int main()

{

char X[20];

char Y[20]; scanf("%s",&\*X);

scanf("%s",&\*Y); int m = strlen(X); int n = strlen(Y); lcs(X, Y, m, n); return 0;

}

#### Dynamic Programming:Longest Common Subsequence

#include<bits/stdc++.h> #include<iostream> #include <string>

using namespace std; int max(int a, int b);

/\* Returns length of LCS for X[0..m-1], Y[0..n-1] \*/ int lcs( char \*X, char \*Y, int m, int n )

{

if (m == 0 || n == 0) return 0;

if (X[m-1] == Y[n-1])

return 1 + lcs(X, Y, m-1, n-1); else

return max(lcs(X, Y, m, n-1), lcs(X, Y, m-1, n));

}

/\* Utility function to get max of 2 integers \*/ int max(int a, int b)

{

return (a > b)? a : b;

}

/\* Driver program to test above function \*/ int main()

{

int m,n; char X[101]; char Y[101]; cin>>X; cin>>Y;

m = strlen(X); n = strlen(Y);

printf("Length of LCS is %d", lcs( X, Y, m, n ) );

return 0;

}

**Longest Common Subsequence** #include<iostream> #include<cstring> #include<cstdlib>

using namespace std;

void lcs( char \*X, char \*Y, int m, int n )

{

int L[m+1][n+1];

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

{

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

{

if (i == 0 || j == 0)

L[i][j] = 0;

else if (X[i-1] == Y[j-1])

L[i][j] = L[i-1][j-1] + 1;

else

}

}

L[i][j] = max(L[i-1][j], L[i][j-1]);

int index = L[m][n]; char lcs[index+1]; lcs[index] = '\0'; int i = m, j = n;

while (i > 0 && j > 0)

{

if (X[i-1] == Y[j-1])

{

lcs[index-1] = X[i-1];

i--; j--; index--;

}

else if (L[i-1][j] > L[i][j-1])

i--;

else

}

j--;

cout << "The Longest Common Subsequence is " << lcs;

}

int main()

{

char X[100],Y[100]; cin>>X>>Y;

int m = strlen(X); int n = strlen(Y); lcs(X, Y, m, n); return 0;

}

**Longest Common Subsequence 1** #include<stdio.h> #include<string.h>

int i,j,m,n,c[20][20];

char x[20],y[20],b[20][20];

void print(int i,int j)

{

if(i==0 || j==0)

return; if(b[i][j]=='c')

{

print(i-1,j-1); printf("%c",x[i-1]);

}

else if(b[i][j]=='u')

print(i-1,j);

else

}

print(i,j-1);

void lcs()

{

m=strlen(x); n=strlen(y); for(i=0;i<=m;i++)

c[i][0]=0;

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

c[0][i]=0;

//c, u and l denotes cross, upward and downward directions respectively for(i=1;i<=m;i++)

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

{

if(x[i-1]==y[j-1])

{

c[i][j]=c[i-1][j-1]+1;

b[i][j]='c';

}

else if(c[i-1][j]>=c[i][j-1])

{

}

else

c[i][j]=c[i-1][j]; b[i][j]='u';

{

}

}

}

int main()

{

c[i][j]=c[i][j-1]; b[i][j]='l';

scanf("%s",x);

scanf("%s",y);

printf("\nThe Longest Common Subsequence is "); lcs();

print(m,n); return 0;

}

**Length of LCS** #include<iostream> #include<cstring> using namespace std; int n,LCS[5010][5010]; char A[5010],B[5010];

int main()

{

cin>>(A+1); cin>>(B+1);

n=strlen(A+1); int i,j,gasit;

gasit=0; for(i=n;i>0;i--)

{

if(gasit==1) LCS[i][n]=1;

else

{

if(A[i]==B[n])

{

gasit=1; LCS[i][n]=1;

}

else

LCS[i][n]=0;

}

}

gasit=0; for(j=n;j>0;j--)

{

if(gasit==1) LCS[n][j]=1;

else

{

if(A[n]==B[j])

{

gasit=1; LCS[n][j]=1;

}

else

LCS[n][j]=0;

}

}

for(i=n-1;i>0;i--)

{

for(j=n-1;j>0;j--)

{

if(A[i]!=B[j]) LCS[i][j]=max(LCS[i+1][j],LCS[i][j+1]);

else

LCS[i][j]=max(1+LCS[i+1][j+1],max(LCS[i+1][j],LCS[i][j+1]));

}

}

cout<<LCS[1][1]<<"\n";

return 0;

}

**Apply KMP** #include<bits/stdc++.h> #define size 100005 using namespace std; int f[size];

void prefix\_function(char p[])

{

f[0]=0;

int j=0;

for(int i=1;i<strlen(p);i++)

{

while(j && p[i]!=p[j])

j=f[i-1]; if(p[i]==p[j])

j++;

f[i]=j;

}

}

int main()

{

char t[size],p[size]; cin>>p;cin>>t; prefix\_function(p);

int j=0,lp=strlen(p),count=0; for(int i=0;i<strlen(t);i++)

{

while(j && t[i]!=p[j])

j=f[j-1]; if(t[i]==p[j])

j++;

if(j==lp)

{

}

}

j=f[j-1]; count++;

printf("%d ",count);

}

# DAA E-Lab

**Session-9**

### Subset of the word

#include <stdio.h>

char string[50], n;

void subset(int, int, int);

int main()

{

int i, len;

// printf("Enter the len of main set : "); scanf("%d", &len);

// printf("Enter the elements of main set : "); scanf("%s", string);

n = len;

// printf("The subsets are :\n"); for (i = 1;i <= n;i++)

subset(0, 0, i);

return 0;

}

/\*Function to find the number of subsets in the given string\*/ void subset(int start, int index, int num\_sub)

{

int i, j;

if (index - start + 1 == num\_sub)

{

if (num\_sub == 1)

{

for (i = 0;i < n;i++) printf("%c\n", string[i]);

}

else

{

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

{

for (i = start;i < index;i++) printf("%c", string[i]);

printf("%c\n", string[j]);

}

if (start != n - num\_sub)

subset(start + 1, start + 1, num\_sub);

}

}

else

{

subset(start, index + 1, num\_sub);

}

}

**Subset of the word 1**

#include <stdio.h>

char string[50], n;

void subset(int, int, int);

int main()

{

int i, len;

// printf("Enter the len of main set : "); scanf("%d", &len);

//printf("Enter the elements of main set : "); scanf("%s", string);

n = len;

// printf("The subsets are :\n"); for (i = 1;i <= n;i++)

subset(0, 0, i);

return 0;

}

/\*Function to find the number of subsets in the given string\*/

void subset(int start, int index, int num\_sub)

{

int i, j;

if (index - start + 1 == num\_sub)

{

if (num\_sub == 1)

{

for (i = 0;i < n;i++) printf("%c\n", string[i]);

}

else

{

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

{

for (i = start;i < index;i++) printf("%c", string[i]);

printf("%c\n", string[j]);

}

if (start != n - num\_sub)

subset(start + 1, start + 1, num\_sub);

}

}

else

{

subset(start, index + 1, num\_sub);

}

}

**List the sub arrays** #include <cstdio> #include <algorithm> #include <numeric> #include <utility> #include <vector>

#define st first #define nd second

using namespace std;

typedef long long ll; typedef pair<int, int> PII; typedef pair<ll, int> PLI;

const int MAXN = 1E5 + 10; PII b[MAXN];

int pre[MAXN], nex[MAXN];

int main(){

int cas;

scanf("%d", &cas);

while (cas--){

int n, m;

scanf("%d%d", &n, &m); for (int i = 1; i <= n; ++i){

scanf("%d", &b[i].st); b[i].nd = i;

pre[i] = i - 1, nex[i] = i + 1;

}

nex[0] = 1, pre[n + 1] = n; sort(b + 1, b + n + 1); vector<PLI> c;

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

int u = b[i].nd;

ll t = (ll)(u - pre[u]) \* (nex[u] - u); nex[pre[u]] = nex[u];

pre[nex[u]] = pre[u];

if (!c.empty() && c.back().nd == b[i].st) c.back().st += t;

else

}

c.push\_back(PLI(t, b[i].st));

reverse(c.begin(), c.end()); for (int i = 1; i < c.size(); ++i)

c[i].st += c[i - 1].st; for (int i = 0; i < m; ++i){

ll t;

scanf("%lld", &t);

printf("%d\n", lower\_bound(c.begin(), c.end(), PLI(t, 0))->nd);

}

}

return 0;

}

**How to choose a subset?**

#include <iostream> #include<math.h> using namespace std;

int main() { int t;

cin>>t; while(t--)

{

long long int n,k,b; cin>>n>>k>>b; long long int cnt=0; n++;

for(long long int i=0;i < (long long int )pow(2,n);i++)

{

if(k == builtin\_popcount (i))

{

//cout<<i<<endl; long long int xr=0;

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

{

if((i & (long long int )pow(2,j)) == (long long int )pow(2,j))

{

xr^=j;

}

}

if(b== builtin\_popcount (xr))

{

cnt++;

}

}

}

cout<<cnt<<endl;

}

return 0;

}

**The Subset Sum**

#include<stdio.h> #include<stdlib.h>

void sumOfSub(int,int,int); static int m=0;

int\*w; int\*x;

int main()

{ int i=0,sum=0,n=0; scanf("%d",&n); w=(int\*)malloc(sizeof(int)\*n+1); x=(int\*)malloc(sizeof(int)\*n+1); for(i=1;i<=n;i++)

{

scanf("%d",&w[i]); sum+=w[i];

x[i]=0;

}

scanf("%d",&m); if(sum<m)

{

exit(1);

}

sumOfSub(0,1,sum); return 0;

}

void sumOfSub(int s,int k,int r)

{

int i=0; x[k]=1;

if(s+w[k]==m)

{

for(i=1;i<=k;i++) printf(" %d",x[i]);

}

else if((s+w[k]+w[k+1])<=m)

{

sumOfSub(s+w[k],k+1,r-w[k]);

}

if((s+r-w[k])>=m&&(s+w[k+1])<=m)

{

x[k]=0;

sumOfSub(s,k+1,r-w[k]);

}

}

**The Castle Gate**

#include <stdio.h>

int lower\_xor\_numbers(int N);

int main(int argc, char const \*argv[]){ int T,N;

int i,j;

scanf("%d\n",&T); for ( i = 0; i < T; ++i){

scanf("%d\n",&N); printf("%d\n",lower\_xor\_numbers(N));

}

return 0;

}

int lower\_xor\_numbers(int N){ int count = 0,i,j;

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

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

if ((i^j)<=N){

count++;

}

}

}

return count;

}

**Possible Permutations** #include<stdio.h> #include<stdlib.h>

void permute(int\* a,int k,int n);

int main()

{

int i,n; int\*a;

scanf("%d",&n); a=(int\*)calloc(n,sizeof(int));

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

permute(a,0,n-1); return 0;

}

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

{ int i,t; if(k==n)

{

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

}else

{

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

{

t=a[k]; a[k]=a[i]; a[i]=t;

permute(a,k+1,n); t=a[k];

a[k]=a[i]; a[i]=t;

}

}

}

**Is it possible?** #include<bits/stdc++.h> using namespace std;

bool isSubsetSum(int set[], int n, int sum)

{

bool subset[n+1][sum+1];

for (int i = 0; i <= n; i++) subset[i][0] = true;

for (int i = 1; i <= sum; i++) subset[0][i] = false;

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

{

for (int j = 1; j <= sum; j++)

{

if(j<set[i-1])

subset[i][j] = subset[i-1][j]; if (j >= set[i-1])

subset[i][j] = subset[i-1][j] || subset[i - 1][j-set[i-1]];

}

}

return subset[n][sum];

}

int main()

{

int set[15]; int sum,i; int n;

cin>>n>>sum; for(i=0;i<n;i++) scanf("%d",&set[i]);

if(isSubsetSum(set,n,sum)==1) printf("Yes");

else

printf("No"); return 0;

}

**Aaryan, Subsequences And Great XOR**

#include <stdio.h>

#define getcx getchar\_unlocked #define putcx putchar\_unlocked inline long long int input()

{

long long int n=0; char ch=getcx();

while( ch < '0' || ch > '9' )

{

ch=getcx();

}

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

{

n = (n<<3)+(n<<1) + ch-'0', ch=getcx();

}

return n;

}

inline void output(long long int n)

{

char a[35];

long long int i=0; do

{

a[i++]=n%10+48;

n=n/10;

}while(n!=0);

--i;

while(i>=0) putcx(a[i--]);

putcx(' ');

}

int main()

{

long long int n; int a,ans=0; n=input(); while(n--)

{

a=input(); ans|=a;

}

output(ans); return 0;

}

**Generating Sequence** #include <iostream> using namespace std;

int main()

{

int T; cin >> T; while(T--){

long long g; int n; cin>>g>>n;

long long g2=g\*2;

// cout<<g2<<" ";

while(--n) cout<<g2<<" "; cout<<g\*3<<'\n';

}

return 0;

}

**Xor is Mad** #include <stdio.h> int main()

{

int i,j,k,l,m,n,t,f[100],a[100]; scanf("%d",&n); for(i=1;i<=n;i++)

{

scanf("%d",&a[i]); f[i]=0;

}

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

{

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

{

if((j|a[i])==(j+a[i])) f[i]=f[i]+1;

}

}

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

return 0;

}

**Jarvis and Lone Integer**

#include <stdio.h>

#define testcase(T) for(scanf("%lld", &T); T; --T)

int i;

long long s, temp, T, N;

int main() { testcase(T) { s = 0;

scanf("%lld", &N); for(i = 0; i < N; i++) { scanf("%lld", &temp); s = s ^ temp;

}

if(s == 0) { printf("-1\n");

} else { printf("%lld\n", s);

}

}

return 0;

}

**Sherlock and XOR** #include <iostream> using namespace std; int main()

{

int t; std::cin>>t; while(t--)

{

long long n; std::cin>>n; long long arr[n];

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

{

std::cin>>arr[i];

}

long long o=0,e=0; for(int i=0;i<n;i++)

{

if(arr[i]%2==0)

{

e++;

}

else

{

o++;

}

}

std::cout<<o\*e<<"\n";

}

return 0;

}

**The Magic**

#include <stdio.h>

int ones(int n)

{

int count = 0; while(n)

{

if(n&1) count++; n>>=1;

}

return count;

}

int main()

{

int t; int n;

scanf("%d",&t); while(t--)

{

scanf("%d",&n);

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

}

return 0;

}

**Localized search engine**

#include <iostream>

int main()

{

int t;

bool letters[26],common; std::string s1,s2; std::cin>>t;

for(int i=0;i<t;i++) {

std::cin>>s1>>s2; common=false; for(int j=0;j<26;j++) {

letters[j]=false;

}

for(int j=0;j<s1.size();j++) {

letters[s1[j]-'a']=true;

}

for(int j=0;j<s2.size();j++) {

if(letters[s2[j]-'a']) {

// std::cout<<"\n";

// std::cout<<"\n";

}

std::cout<<"Yes"<<std::endl;

common = true; break;

}

if(!common) {

std::cout<<"No"<<std::endl;

}

}

}

**Aaryan, Subsequences And Great XOR**

#include <stdio.h>

#define getcx getchar\_unlocked #define putcx putchar\_unlocked inline long long int input()

{

long long int n=0; char ch=getcx();

while( ch < '0' || ch > '9' )

{

ch=getcx();

}

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

{

n = (n<<3)+(n<<1) + ch-'0', ch=getcx();

}

return n;

}

inline void output(long long int n)

{

char a[35];

long long int i=0; do

{

a[i++]=n%10+48;

n=n/10;

}while(n!=0);

--i;

while(i>=0) putcx(a[i--]);

putcx(' ');

}

int main()

{

long long int n; int a,ans=0; n=input(); while(n--)

{

a=input(); ans|=a;

}

output(ans); return 0;

}

**Monk and his Friend**

#include <stdio.h>

int main()

{

int t,count; scanf("%d",&t); while(t--)

{

long long int a,b,k; scanf("%lld %lld",&a,&b); if(a==b)

printf("0\n"); else

{

k=a^b;count=0; while( k )

{

k = k&(k-1); count++;

}

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

}

}

return 0;

}

**Monk's Choice of Numbers** #include <iostream> #include <queue>

using namespace std;

int findones(int n)

{

int count = 0; while(n)

{

if(n&1) count++; n>>=1;

}

return count;

}

int main()

{

int t;

cin>>t; while(t--)

{

priority\_queue<int> pq; int n,k;

cin>>n>>k; while(n--)

{

int x; cin>>x;

pq.push(findones(x));

}

int sum = 0; while(k--)

{

sum += pq.top(); pq.pop();

}

cout<<sum<<endl;

}

return 0;

}

**Monk and Tasks** #include<bits/stdc++.h> using namespace std;

int countOnes(int num) { int cnt=0;

while(num) {

num &= num-1; cnt++;

}

return cnt;

}

int main()

{

int t,i,j,temp,k,n; cin>>t; for(i=0;i<t;i++) {

cin>>n;

int vec[n]; vector<pair<int,int>> mv; for(j=0;j<n;j++) {

cin>>temp; vec[j]=temp;

}

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

k = countOnes(vec[j]); mv.push\_back(make\_pair(k,j));

}

sort(mv.begin() , mv.end()); for(j=0;j<n;j++) {

cout<<vec[mv[j].second]<<" ";

}

cout<<endl;

}

}

**Subset AND** #include<bits/stdc++.h> using namespace std; int main(void) {

int t; cin>>t; while(t--){

int n,m; cin>>m>>n; int a[n];

for(int i=0;i<n;i++){ cin>>a[i]; m&=a[i];

}

if(m==0){ cout<<"Yes\n";

}

else{

cout<<"No\n";

}

}

return 0;

}

**Micro and Binary Strings** #include <bits/stdc++.h> using namespace std;

int main()

{

ios\_base::sync\_with\_stdio(false); cin.tie(NULL);

int q,x;

char a[100000]; cin>>q;

while(q--)

{

int count=0; cin>>x; cin>>a;

for(int i=0;i<x;i++)

{

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

++count;

}

cout<<count<<'\n';

}

return 0;

}

**Subset XOR**

#include <iostream> using namespace std;

int findXOR(int Set[], int n)

{

if (n == 1) return Set[0];

else

return 0;

}

int main()

{

int t; cin>>t;

for(int i=1;i<=t;i++)

{

int size; cin>>size; int Set[size];

for(int j=1;j<size;j++) cin>>Set[i];

cout << findXOR(Set, size)<<"\n";

}return 0;

}

**Benny And Subsets** #include<bits/stdc++.h> using namespace std; typedef long long int lli; int n,req;

vector<int> v; vector<int> arr; lli xr[100000000];

lli dp[1001][4100];

#define mod 10000007

int bitmasking(vector<int> v)

{

int len=v.size(); int msk=1<<len;

for(int i=0;i<msk;i++)

{

int x=0;

for(int j=0;j<len;j++)

{

if(i&(1<<j)) x=x xor v[j];

}

xr[x]=(xr[x]+1)%mod;

}

}

int main()

{

cin>>n>>req; for(int i=0;i<n;i++)

{

int a; cin>>a;

if(a>(1<<10)) v.push\_back(a); else arr.push\_back(a);

}

memset(dp,0,sizeof dp); n=arr.size();

bitmasking(v); dp[0][0]=1;

int i;

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

{

int val=arr[i-1];

for(int j=0;j<=2048;j++)

{

dp[i][j]=dp[i-1][j];

}

for(int j=0;j<=2048;j++)

{

if((j xor val)<=2048) dp[i][j]=(dp[i][j]+dp[i-1][j xor val])%mod;

}

}

lli ans=0;

for(int i=0;i<=2048;i++)

{

ans =(ans+(1ll\*dp[n][i]\*(xr[i xor req])%mod))%mod;

}

cout<<ans<<endl;

}

**Subset Sum** #include<stdio.h> int main()

{

int t; scanf("%d",&t); while(t--)

{

int n,i; scanf("%d",&n); long long int ans=1; for(i=0;i<n-1;i++)

ans=ans<<1; ans=ans\*(n\*(n+1))/2; printf("%lld\n",ans);

}

return 0;

}

**Sherlock and Coprime Subset** #include<bits/stdc++.h> using namespace std;

int dp[55][1<<16];

int arr[55]; vector<int> v; int pr[55];

int n; int sz;

int check(int mask,int num)

{

int reply=0,f=0; for(int i=0;i<sz;i++)

{

int kk=v[i];

if((mask & (1<<i)) && num%kk==0)

{

f=1;

break;

}

else if(num%kk==0)

{

reply= reply | (1<<i);

}

}

if(f==1) return -1; else return reply;

}

int primes()

{

pr[1]=0;

for(int i=2;i<=55;i++)

{

if(!pr[i])

for(int j=2;j\*i<=50;j++)

{

pr[j\*i]=1;

}

}

int j=0;

for(int i=2;i<=50;i++)

{

if(!pr[i])

{

v.push\_back(i);

}

}

sz=v.size();

}

int solve(int mask,int pos)

{

if(pos>=n) return 0;

if(dp[pos][mask]!=-1) return dp[pos][mask]; else

{

int ret=0;

int num=arr[pos];

int reply=check(mask,num); if(reply!=-1)

{

int nm=mask | reply ; ret=max(ret,solve(nm ,pos+1)+1) ;

}

ret=max(ret,solve(mask,pos+1)); dp[pos][mask]=ret;

return ret;

}

}

int main()

{

primes(); int t; cin>>t; while(t--)

{

memset(dp,-1,sizeof dp); cin>>n;

int ans=0; int j=0;

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

{

int a; scanf("%d",&a); if(a==1) ans++; else

arr[j++]=a;

}

n=j;

ans+=solve(0,0); cout<<ans<<endl;

}

}

**Panda and XOR**

#include <stdio.h>

long long mod = 1000000007; long long ret(long long n){

if(n==0||n==1)return 0; n%=mod;

return ((n\*(n-1))/2);

}

int main()

{

long long n,i,j,myind,no,total=0; scanf("%lld",&n);

long long dp[128][2],arr[n+1];

for(i=0;i<2;i++){ for(j=0;j<=127;j++){

dp[j][i] = 0;

}

}

for(i=1;i<=n;i++){ scanf("%lld",&arr[i]); total^=arr[i];

}

for(i=1;i<=n;i++){ no = arr[i];

for(j=0;j<=127;j++){ myind = arr[i]^j;

dp[myind][1] = (dp[myind][0] + dp[j][0])%mod;

}

dp[no][1] = (dp[no][1]+1)%mod; for(j=0;j<=127;j++){

dp[j][0] = dp[j][1];

}

//printf("%lld %lld %lld %lld \n",dp[0][1],dp[1][1],dp[2][1],dp[3][1]);

}

long long sum= ret(dp[0][1]); for(i=1;i<=127;i++){

sum = (sum + ret(dp[i][1]))%mod;

}

printf("%lld",sum); return 0;

}

**Sum of Digits** #include <stdio.h> #define S 1000000007

typedef long long int ll;

//long long int exponentiation(ll x,ll n)

//{}

ll compare (const void \* a, const void \* b)

{

return ( \*(ll\*)a - \*(ll\*)b );

}

int main()

{

ll n,i;

ll arr[100005]; int arr1[100005]; scanf("%lld",&n); arr[0]=1;

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

{

arr[i]=(arr[i-1]\*2)%S;

}

ll no,j,sum=0; for(i=0;i<n;i++)

{

scanf("%lld",&no); for( j=no;j>0;j=j/10) sum+=j%10; arr1[i]=sum; sum=0;

}

qsort (arr1, n, sizeof(int), compare);

//qsort(); ll count=0;

for(i=n-1;i>=0;i--)

{

count=(count%S+(arr1[i]\*arr[i])%S)%S;

}

printf("%lld",count); return 0;

}

**Power Set Game** #include <stdio.h> #include <math.h>

#define MOD 1000000007

long long lpower(int a, int n)

{

int i = 0;

long long an = 1; while(n > 0)

{

an = (an \* a) % MOD; n -= 1;

}

return an;

}

int main()

{

int T = 0;

scanf ("%d", &T); int i = 0;

int N = 0;

unsigned long long s;

while (i < T)

{

i++;

scanf("%d", &N);

//printf("%d\n", lpower(2,2));

s = lpower(4, N) - lpower(2,N); s = (s+MOD) % MOD;

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

}

return 0;

}

**Subset Sum Problem** #include <stdio.h> #include <stdbool.h>

bool isSubsetSum(int set[], int n, int sum)

{

if (sum == 0) return true;

if (n == 0 && sum != 0) return false;

if (set[n-1] > sum)

return isSubsetSum(set, n-1, sum);

return isSubsetSum(set, n-1, sum) ||

isSubsetSum(set, n-1, sum-set[n-1]);

}

int main()

{

int set[100],i,n; int sum = 9; scanf("%d",&n); for (i=1;i<=n;i++)

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

if (isSubsetSum(set, n, sum) == true) printf("Found a subset with given sum");

else

printf("No subset with given sum"); return 0;

}

**Chandu and Consecutive Letters**

#include <stdio.h> int main()

{

int n=0,i=0,j=0,k=0; char a[30]; scanf("%d",&n); for(;i<n;i++)

{

scanf("%s",a); for(;a[k]!='\0';k++)

{

if(a[k]!=a[k-1])

{

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

}

}

k=0;

printf("\n");

}

return 0;

}

**DAA E-Lab**

**Session-10**

**MINMAX 1**

#include <stdio.h> #include <string.h> #include <math.h> #include <stdlib.h>

int compare2units(const void\*a,const void\*b){ return (\*(unsigned int\*)a - \*(unsigned int\*)b);

}

int main()

{

int n,k,i;

unsigned int x[100001]; unsigned int j,minunfair; if(scanf("%d",&n)!=1)return 1;

if(scanf("%d",&k)!=1)return 1;

for(i=0;i<n;i++) if (scanf("%d",x+i)!=1) return 1; if(k==1) return 0; qsort(x,n,sizeof(int),compare2units); minunfair=0x7fffffff;

i=0;

j=k-1; while(j<n){

if (x[j]-x[i]<minunfair) minunfair=x[j]-x[i]; i++;

j++;

}

printf("%u\n",minunfair); return 0;

}

**MINMAX 2**

#include<stdio.h> int a[50],max,min;

void find(int i,int n){ int mid,max1,min1;

if(i==n)

max=min=a[i]; else if(i==n) if(a[i]>=a[n]){

max=a[n];

min=a[i];

}

else{ max=a[i];

min=a[n];

}

else{ mid=(i+n)/2; find(i,mid); max1=max; min1=min; find(mid+1,n); if(max<max1)

max=max1; if(min>min1)

min=min1;

}

}

int main(){ int i,n;

scanf("%d",&n); for(i=0;i<n;i++)

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

max=min=a[i]; find(0,n-1);

printf("Minimum : %d",min); printf("\nMaximum : %d",max);

return 0;

}

**RANDOMIZED ALGORITHMS 1**

#include <iostream> using namespace std; int main()

{

int a,b; cin>>a>>b; if(a\*b>0)

cout<<"Signs are not opposite"; else

cout<<"Signs are opposite"; return 0;

}

**RANDOMIZED ALGORITHMS 2**

#include <stdio.h>

unsigned int countSetBitsUtil(unsigned int x);

unsigned int countSetBits(unsigned int n)

{

int bitCount = 0; // initialize the result int i;

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

bitCount += countSetBitsUtil(i);

return bitCount;

}

unsigned int countSetBitsUtil(unsigned int x)

{

if (x <= 0) return 0;

return (x % 2 == 0 ? 0 : 1) + countSetBitsUtil(x / 2);

}

// Driver program to test above functions int main()

{

// int n = 8; int n;

scanf("%d",&n);

printf("Total set bit count is %d", countSetBits(n)); return 0;

}

**RANDOMIZED ALGORITHMS 3**

#include <stdio.h> #include <string.h>

void swap(char \*x, char \*y)

{

char temp; temp = \*x;

\*x = \*y;

\*y = temp;

}

void permute(char \*a, int l, int r)

{

int i;

if (l == r) printf("%s\n", a); else

{

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

{

swap((a+l), (a+i)); permute(a, l+1, r); swap((a+l), (a+i));

}

}

}

int main()

{

char str[100]; scanf("%s",str); int n = strlen(str);

permute(str, 0, n-1);

return 0;

}

**RANDOMIZED ALGORITHMS 4**

#define MAX\_POINT 3

#define ARR\_SIZE 100 #include<stdio.h>

void printArray(int arr[], int arr\_size);

void printCompositions(int n,int i)

{

static int arr[ARR\_SIZE]; if(n==0)

{

printArray(arr,i);

}

else if(n>0)

{

int k; for(k=1;k<=MAX\_POINT;k++)

{

arr[i]=k; printCompositions(n-k,i+1);

}

}

}

void printArray(int arr[], int arr\_size)

{

int i; for(i=0;i<arr\_size;i++)

{

if(i!=arr\_size-1) printf("%d ",arr[i]); else

{

printf("%d",arr[i]); printf("\n");

}

}

// printf("\n");

}

int main()

{

int n;// = 5; scanf("%d",&n);

printf("Differnt compositions formed by 1, 2 and 3 of %d are",n); printf("\n");

// Differnt compositions formed by 1, 2 and 3 of 4 are

printCompositions(n,0); return 0;

}

**RANDOMIZED ALGORITHMS 5**

#include<iostream> using namespace std;

typedef unsigned int uint\_t; uint\_t snoob(uint\_t x)

{

uint\_t rightOne;

uint\_t nextHigherOneBit; uint\_t rightOnesPattern; uint\_t next = 0;

if(x)

{

rightOne = x & -(signed)x; nextHigherOneBit = x + rightOne; rightOnesPattern = x ^ nextHigherOneBit;

rightOnesPattern = (rightOnesPattern)/rightOne; rightOnesPattern >>= 2;

next = nextHigherOneBit | rightOnesPattern;

}

return next;

}

int main()

{

int x; cin>>x;

cout<<snoob(x); getchar(); return 0;

}

**RANDOMIZED ALGORITHMS 6**

#include<stdio.h>

int addOne(int x)

{

int m = 1;

/\* Flip all the set bits until we find a 0 \*/ while( x & m )

{

x = x^m; m <<= 1;

}

/\* flip the rightmost 0 bit \*/ x = x^m;

return x;

}

/\* Driver program to test above functions\*/ int main()

{

int x; scanf("%d",&x);

printf("%d", addOne(x));

// getchar(); return 0;

}

**RANDOMIZED ALGORITHMS 7**

#include <stdio.h>

int multiplyWith3Point5(int x)

{

return (x<<1) + x + (x>>1);

}

/\* Driver program to test above functions\*/ int main()

{

int x; scanf("%d",&x);

printf("%d", multiplyWith3Point5(x)); getchar();

return 0;

}

**RANDOMIZED ALGORITHMS 8**

#include <iostream> using namespace std;

int Add(int x, int y)

{

while (y != 0)

{

int carry = x & y; x = x ^ y;

y = carry << 1;

}

return x;

}

int main()

{

int a,b; cin>>a>>b; cout<<Add(a,b);

return 0;

}