CoU Linear Algebra

Bonusree ,Shimu , Lameya

December 2022

1 NUMBER THEORY

1.1 SIEVE

```
void seive()
{
   prime[1]=1;
   for(long long int i=2;i<=lim;i+=2)
   prime[i]=2;
   nprime.push_back(2);
   lli x=0;
   for(long long int  i=3;i<=lim;i+=2)
   {
    if(prime[i]==0)
    {
      nprime.push_back(i);
        prime[i]=i;
        for(long long int  j=i*i;j<=lim;j+=i*2)
            prime[j]=i;
    }
   }
}
```

1.2 Segmented Sieve

```
    void segSieve (ll l, ll r) {
    bool isPrime[r-l+1];
    for (int i = 0; i < r - l + 1; ++i) isPrime[i] = true;
    if (l == 1) isPrime[0] = false;
    for (int i = 0; primes[i]*primes[i] <= r; ++i) {
        int currentPrime = primes[i];
        ll base = (l/currentPrime)*currentPrime;
        if (base < l) base += currentPrime;
        for (ll j = base; j <= r; j += currentPrime) {
            isPrime[j-l] = false;
        }
        if (base == currentPrime) isPrime[base-l] = true;
    }
    for (int i = 0; i < r - l + 1; ++i) {
        if (isPrime[i]) cout << (i+l) << endl;
    }
    puts("");
}
```

1.3 Sum of Divisor

```
 long long SumOfDivisor(long long n)
{
 long long ans=1;
 for(long long i=0; prime[i]*prime[i]<=n; i++)
 {long long sum=0,p=1;
    while(n%prime[i]==0)
   {
        n/=prime[i];   p*=prime[i];   sum+=p;
    }
    ans*=(sum+1);
 }
 if(n>1)
 ans*=(n+1);
 return ans;}
```

1.4 BigMOd

```
long long bigmod(long long n, long long p, long long m)
{
 if (p == 0)
 return 1;
 long long x = bigmod(n, p >> 1, m);
 x = (x * x) % m;
 if (p & 1)
 x = (x * n)% m;
 return x;
}
```

1.5 NOD

```
long long NumberOfDivisor(long long n)
{
 long long ans=1;
 for(long long i=0; prime[i]*prime[i]<=n; i++)
 {
   long long counter=0;
   while(n%prime[i]==0)
     {
        n/=prime[i];
        counter++;
    }
 ans*=(counter+1);
 }
 if(n>1)
 ans*=2;
 return ans;
}
```

2 Geometry

2.1 Triangle

Circumradius,

$$r = \frac{abc}{(a+b+c)(b+c-a)(c+a-b)(a+b-c)} \tag{1}$$

$$r = \frac{abc}{4 \times AreaOfTriangle} \tag{2}$$

Incircle radius,

$$r = \frac{1}{2 \times ra} + \frac{1}{2 \times rb} + \frac{1}{2 \times rc} = AreaOfTriangle \tag{3}$$

Excircle radius(if the circle is tangent to side of the triangle

$$r = Incircleradius \times \frac{a+b+c}{b+c-a} \tag{4}$$

$$r = 2 \times \frac{AreaOfTriangle}{b+c-a} \tag{5}$$

$$Heron'sFormula = \sqrt{s(s-a)(s-b)(s-c)} \tag{6}$$

$$Heron'sFormula = \sqrt{s(s-a)(s-b)(s-c)} \tag{7}$$

$$SineRule = \frac{a}{sinA} = \frac{b}{sinB} = \frac{c}{sinC} = 2R \tag{8}$$

$$CosineRule = a^2 = b^2 + c^2 - 2bcCosA \tag{9}$$

2.2 Circle

$$ArcLength, s = r\theta(angleinradian) \tag{10}$$

$$SectorArea = \frac{\theta}{2} \times r^2(angleinradian) \tag{11}$$

$$chordlength, d = 2 \times r \times sin(\frac{\theta}{2})(angleinradian) \tag{12}$$

$$chordlength, d = 2 \times \sqrt{r^2 - x^2}(x = perpendiculardistancefromthecentretochord) \tag{13}$$

$$OutsideOneAnother, C_1C_2 > r_1 + r_2 \tag{14}$$

$$TouchingEexternally, C_1C_2 = r_1 + r_2 \tag{15}$$

$$IntersectingAt2Points, |r_1 + r_2| < C_1C_2 < r_1 + r_2 \tag{16}$$

$$TouchingIinternally, C_1C_2 = |r_1 - r_2| \tag{17}$$

$$OneInsideTheOther, C_1C_2 < |r_1 - r_2| \tag{18}$$

2.3 Others

$$Cube, area = 6a^2 \tag{19}$$

$$cube, volume = a^3 \tag{20}$$

$$cylinder, area = 2\pi rh + 2\pi r^2 \tag{21}$$

$$cylinder, volume = \pi r^2 h \tag{22}$$

$$Cone, area = \pi rl \tag{23}$$

$$Cone, volume = \frac{1}{3}\pi r^2 h \tag{24}$$

$$sphere, area = 4\pi r^2 \tag{25}$$

$$sphere, volume = \frac{4}{3}\pi r^3 \tag{26}$$

3 Segment

3.1 Segment Tree

```
void init(lli node, lli start, lli end)
{
  if(start==end)
  {
    tree[node]=arr[start];     return;
   }
  lli mid=(start+end)/2, left=node*2, right=node*2+1;
  init(left,start,mid);
  init(right,mid+1,end);
  tree[node]=tree[left]+tree[right];
}
void update(lli node,lli start,lli end,lli i,lli val)
{
  if(i<start||i>end)
  return;
  else if(i==start&&i==end)
  {
    tree[node]=val;
    return;
  }
  lli mid=(start+end)/2, left=node*2,  right=node*2+1;
  update(left,start,mid,i,val);
  update(right,mid+1,end,i,val);
  tree[node]=tree[right]+tree[left];
}
lli query(lli node,lli start,lli end,lli l,lli r)
{
  lli m=0;
  if(r<start||l>end)
  return m;
  else if(l<=start&&r>=end)
  return tree[node];
  lli mid=(start+end)/2, left=node*2, right=node*2+1;
  lli q1=query(left,start,mid,l,r);
  lli q2=query(right,mid+1,end,l,r);
  return q1+q2;
}
```

3.2 Lazy propogation

```
    lli ar[mx];
struct info
{
    lli prop,sum;
}tree[mx*3];
void init(lli pos,lli suru,lli ses)
{
    if(suru==ses)
    {
        tree[pos].sum=ar[ses];
        tree[pos].prop=0;
   }
    lli bam=pos*2, dan=pos*2+1, mid=(suru+ses)/2 ;
    init(bam,suru,mid);
    init(dan,mid+1,ses);
    tree[pos].sum=tree[bam].sum+tree[dan].sum;
    tree[pos].prop=0;
}
lli query(lli pos,lli suru,lli ses,lli f,lli l,lli carry=0 )
{
    if(suru>l||ses<f)
    return 0;
    else if(suru>=f&&ses<=l)
    {
        return tree[pos].sum+(l-f+1)*carry;
    }
    lli bam=pos*2,dan=pos*2+1,mid=(suru+ses)/2;
   lli q1= query(bam,suru,mid,f,l,tree[pos].prop);
    lli q2=query(dan,mid+1,ses,f,l,tree[pos].prop);
    return q1+q2;
}
void update(lli pos,lli suru,lli ses,lli f,lli l,lli val)
{
    if(suru>l||ses<f)
    return ;
    else if(suru>=f&&ses<=l)
    {
        tree[pos].sum+=(val*(l-f+1));
        tree[pos].prop=val;
        return;
    }
    lli bam=pos*2;
    lli dan=pos*2+1;
    lli mid=(suru+ses)/2;
    update(bam,suru,mid,f,l,val);
    update(dan,mid+1,ses,f,l,val);
    tree[pos].sum=tree[bam].sum+tree[dan].sum+tree[pos].prop;
}
```

3.3 Maximum Subarray

```
    vector<int> maxCrossingSum(int arr[], int l, int m, int h){
    int sum = 0,index_l,index_r;
    int left_sum = INT_MIN;
    for (int i = m; i >= l; i--) {
            sum = sum + arr[i];
            if (sum > left_sum)
                left_sum = sum,index_l=i;
        }
    sum = 0;
    int right_sum = INT_MIN;
    for (int i = m+1; i <= h; i++){
            sum = sum + arr[i];
            if (sum > right_sum)
                right_sum = sum,index_r=i;
        }
    vector <int> all{left_sum + right_sum,index_l,index_r};
    return all;
}
vector<int> maxSubArraySum(int arr[], int l, int h){
    if (l == h) {
            vector<int> all{arr[l],l,h};
            return all;
        }
    int m = (l + h)/2;
    return max(maxSubArraySum(arr, l, m),
               max(maxSubArraySum(arr, m+1, h),
                   maxCrossingSum(arr, l, m, h)));
}
```

3.4 LCS

```
void lcs(string s,string s1)
{
    lli l1=s.length(), l2=s1.length(), i,j,k;
    for(i=0;i<=l1;i++)
    {
        for(j=0;j<=l2;j++)
        {
            if(i==0||j==0)arr[i][j]=0;
            else
            {
                if(s[i-1]==s1[j-1])
                {
                    arr[i][j]=arr[i-1][j-1]+1;
                    br[i][j]=1;
                }
                else if(arr[i-1][j]>arr[i][j-1])
                {
                    arr[i][j]=arr[i-1][j];
                    br[i][j]=2;
                }
                else
                {
                    arr[i][j]=arr[i][j-1];
                    br[i][j]=3;
                }
            }
        }
    }
    for(i=0;i<=l1;i++)
    {
        for(j=0;j<=l2;j++)
        cout<<arr[i][j]<<" ";
        cout<<endl;
    }
    for(i=0;i<=l1;i++)
    {
        for(j=0;j<=l2;j++)
        cout<<br[i][j]<<" ";
        cout<<endl;
    }
    string ans;
    i=l1;
    j=l2;

    while(i>0&&j>0)
    {
        if(br[i][j]==1)
        {
            ans+=s[i-1];
            i--;j--;
        }
        else if(br[i][j]==2)
        i--;
        else
        j--;
    }
    reverse(ans.begin(),ans.end());
    cout<<ans<<endl;
}
```

3.5 Merge sort tree

```
lli ar[ma];
vector<lli> tre[ma*3];
void merge(vector<lli>&v1,vector<lli>&v2,vector<lli>&v)
{
    lli i=0,j=0,n=v1.size(),m=v2.size();
    while(i<n&&j<m)
   {
        if(v1[i]<=v2[j])
        {
            v.push_back(v1[i]);  i++;
        }
        else
       {
            v.push_back(v2[j]);j++;
       }
    }
    while(j<m)
    {
        v.push_back(v2[j]);j++;
    }
    while(i<n)
    {
        v.push_back(v1[i]);i++;
   }
}
void mt(lli b,lli e,lli n)
{
    if(b>e)
    return;
    if(b==e)
    {
      tre[n].push_back(ar[b]);return ;
    }
    lli mid=(b+e)/2;
    mt(b,mid,2*n);
    mt(mid+1,e,2*n+1);
    merge(tre[n*2],tre[n*2+1],tre[n]);
}
lli query(lli n,lli b,lli e,lli l,lli r,lli k)
{
    if(r<b||e<l||b>e)
    return 0;
    if(r>=e&&l<=b)
    {
        lli m=upper_bound(tre[n].begin(),tre[n].end(),k)-tre[n].begin();
        return tre[n].size()-m;
    }
    lli mid=(b+e)/2;
    lli q1=query(n*2,b,mid,l,r,k);
    lli q2=query(n*2+1,mid+1,e,l,r,k);
    return q1+q2;
}
```

3.6 KMP

```
const long long int lim = 10e5 + 3;
lli arr[lim], tree[lim * 3];
vector<lli> createTempArray(string ptr)
{
  lli l=ptr.size(),index=0;
  vector<lli>lps(l+1);
  for(lli i=1;i<l;)
  {
    if(ptr[i]==ptr[index])
    {
      lps[i]=index+1;
      index++;
      i++;
    }
    else{
      if(index!=0)
      index=lps[index-1];
      else
      lps[i]=index,i++;
    }
  cout<<i<<endl;
  }
 return lps;
 }
void kmp(string str,string ptr)
{
  vector<lli>lps=createTempArray(ptr);
  lli i=0,j=0,cnt=0;
 while(i<str.size())
  {
    if(str[i]==ptr[j])
    {
      i++;j++;
    }
    else
    {
      if(j!=0)   j=lps[j-1];
      else    i++;
    }
   if(j==ptr.size())cnt++;
  }
  cout<<cnt<<endl;
```

```
   }
```

4 GRAPH

4.1 BFS

```
    vector<vector<int>> adj;
int n; // number of nodes
int s; // source vertex

queue<int> q;
vector<bool> used(n);
vector<int> d(n), p(n);

q.push(s);
used[s] = true;
p[s] = -1;
while (!q.empty()) {
    int v = q.front();
    q.pop();
    for (int u : adj[v]) {
        if (!used[u]) {
            used[u] = true;
            q.push(u);
            d[u] = d[v] + 1;
            p[u] = v;
        }
    }
}
```

4.1.1 FindPath

```
    if (!used[u]) {
    cout << "No path!";}
    else {
    vector<int> path;
    for (int v = u; v != -1; v = p[v])
        path.push_back(v);
    reverse(path.begin(), path.end());
    cout << "Path: ";
    for (int v : path)
        cout << v << " ";
}
```

4.1.2 Floyedfulkerson

```
long long n,i,j,cc=0,m,k;
long long adj[100][100];
long long path[100][100];
void floyed_Warshal()
{
 for(k=1;k<=n;k++){
   for(i=1;i<=n;i++){
     for(j=1;j<=n;j++)
        if(adj[i][k]+adj[k][j]<adj[i][j]){
            adj[i][j]=adj[i][k]+adj[k][j];
            path[i][j]=path[i][k];
        }
    }
}
```

4.2 DFS

4.2.1 Diameter

```
 void dfs_downpath(lli n, lli par)
{
  for (auto it : vc[n])
  { if (it != par)
    {dfs_downpath(it, n);
     downPath[n]=max(downPath[n],1+downPath[it]);
    }
  }
}
void diameter_dfs(lli n,lli par)
{
  vector<lli>children;
  lli ans=0;
  for(auto it:vc[n])
  {if(it!=par)
    {
      diameter_dfs(it,n);
      children.push_back(downPath[it]);
      ans=max(ans,diametre[it]);
    }
  }
  lli numOFchild=children.size();
  if(numOFchild==0)
  diametre[n]=0;
  else if(numOFchild==1)
  diameter[n]=children[0]+1;
  else
  {
    sort(children.rbegin(),children.rend());
    diameter[n]=2+children[0]+children[1];
  }
  diameter[n]=max(ans,diameter[n]);
}
```

4.2.2 DISTANCE

```
 void distance_dfs(lli n,lli par,lli par_dis)
{
  vector<pair<lli,lli>>children;
  lli ans=0;
  for(auto it:vc[n])
  {
    if(it!=par)
    {
      children.push_back({downPath[it],it});
    }
  }
 sort(children.rbegin(),children.rend());
 for(lli i=0;i<children.size();i++)
 {
   if(i==0)
   {
    if(children.size()>1)
    distance_dfs(children[i].second,n,max(children[1].first+2,par_dis+1));
    else
    distance_dfs(children[i].second,n,par_dis+1);
   }
   else
distance_dfs(children[i].second,n,max(children[0].first+2,par_dis+1));
 }
 dis[n]=max(par_dis,downPath[n]);
}
```

5 MACROS

```
#define pi acos(-1.0)
#define dtor(x) (pi*x)/180.0
#define rtod(x) (x*180.0)/p
ios_base::sync_with_stdio(0);
cin.tie(0);
cout.tie(0);
#include <bits/stdc++.h>
using namespace std;
#define lli long long int
```