Answer to the question no-01

```
#include <bits/stdc++.h>
using namespace std;
#define ll long long
const int N=1e3+5;
int n,m;
int depth[N][N];
int dx[]= {0,0,1,-1};
int dy[]= {1,-1,0,0};
int visited[N][N];
int sum;
bool is_inside(int x,int y)
{
   if(x>0&&x<=n && y>0 && y<=m)
   {
      return true;
   }
   return false;
}
int DFS(int x,int y)
{
   if(depth[x][y]==0||!is_inside(x,y)||visited[x][y]==1)
   {
      return 0;
   }
   visited[x][y]=1;
   return DFS(x-1,y)+DFS(x,y-1)+DFS(x+1,y)+DFS(x,y+1)+depth[x][y];
}
int main()
{
   int t;
   cin>>t;
   while(t--)
   {
      int mx=0;
      cin>>n>>m;
      for(int i=1; i<=n; i++)
```

```
        {
            for(int j=1; j<=m; j++)
            {
                cin>>depth[i][j];
            }
        }
        for(int i=1; i<=n; i++)
        {
            for(int j=1; j<=m; j++)
            {
                if(visited[i][j]==0)
                {
                    sum=0;
                    mx=max(mx,DFS(i,j));
                }
            }
        }
        cout<<mx<<endl;
         for(int i=1; i<=n; i++)
        {
            for(int j=1; j<=m; j++)
            {
                visited[i][j]=0;
            }
        }
    }
}
```

<u>Answer to the question no-02</u>

```
#include<bits/stdc++.h>
using namespace std;
const int N=1e5+5;
const long long int INF=1e18;
vector<pair<int,int>>adj_list[N];
long long int d[N];
int visited[N],parent[N];
int nodes,edges;
```

```
void Dijkstra(int src)
{
   for(int i=1;i<=nodes;i++)
   {
      d[i]=INF;
   }
   d[src]=0;
   priority_queue<pair<long long int,int>>pq;
   pq.push({0,src});
   while(!pq.empty())
   {
      pair<long long int ,int>head=pq.top();
      pq.pop();
      int selected_node=head.second;
      if(visited[selected_node])
      {
         continue;
      }
      visited[selected_node]=1;
      for(auto adj_entry:adj_list[selected_node])
      {
         int adj_node=adj_entry.first;
         int edge_cst=adj_entry.second;
         if(d[selected_node]+edge_cst<d[adj_node])
         {
            d[adj_node]=d[selected_node]+edge_cst;
            parent[adj_node]=selected_node;
            pq.push({-d[adj_node],adj_node});
         }
      }
   }
}
int main()
{
   memset(visited,0,sizeof(visited));
   cin>>nodes>>edges;
   for(int i=1;i<=edges;i++)
```

```
    {
        int u,v,w;
        cin>>u>>v>>w;
        adj_list[u].push_back({v,w});
    }

    int src,dst;
    cin>>dst>>src;
    Dijkstra(src);
    if(visited[dst]==0||src==dst)
    {
        cout<<-1;
        return 0;
    }
    cout<<d[dst];
}
```

<u>Answer to the question no-03</u>

```
#include<bits/stdc++.h>
using namespace std;
#define ll long long int
const int N=1e6+5;
const ll INF=1e10;
const ll mod=1e9+7;
ll dp[N];
vector<ll>coin;
int x;
ll min_coin(int n)
{
    if(n==0)
    {
        return 1;
    }
    if(dp[n]!=0)
    {
        return dp[n];
    }
    for(int i=0;i<x;i++)
    {
        if(n-coin[i]>=0)
        {
            dp[n]+=min_coin(n-coin[i]);
            dp[n]%=mod;
        }
    }
    return dp[n];
}
int main()
{
    int a;
    cin>>x>>a;
    coin.resize(x+1);
    for(int i=0;i<x;i++)
    {
        cin>>coin[i];
    }
    ll ans=min_coin(a);
    cout<<ans;
}
```

Answer to the question no-04

```
#include<bits/stdc++.h>
using namespace std;
#define ll long long int
pair<int, int>index(vector<int>nums, int target)
{
    unordered_map<int, int>mp;
    for (int i = 0; i < nums.size(); i++)
    {
        if(nums[i]==0)
        {
            continue;
        }
        int a= target - nums[i];
        if (mp.count(a))
        {
            return {mp[a], i};
        }
        mp[nums[i]] = i;
    }
    return {-1, -1};
}
int main()
{
    int n, x;
    cin >> n >> x;

    vector<int> nums(n);
    for (int i = 0; i < n; i++)
    {
        int a;
        cin>>a;
        if(a>x)
        {
            nums[i]=0;
            continue;
        }
        nums[i]=a;
    }

    pair<int, int> result = index(nums, x);

    if (result.first == -1 && result.second == -1)
    {
        cout << "IMPOSSIBLE" << endl;
    }
    else
    {
        cout << result.first+1 << " " << result.second+1 << endl;
    }
    return 0;
}
```

Answer to the question no-05

```
#include<bits/stdc++.h>
using namespace std;
#define ll long long int
int value[N];
int main()
{
    int n;
    cin>>n;
    for(int i=1;i<=n;i++)
    {
        cin>>value[i];
    }
    set<int>money_sum;
    money_sum.insert(0);
    for(int i=1;i<=n;i++)
    {
        vector<int>temp;
        for(int sum:money_sum)
        {
            temp.push_back(value[i]+sum);
        }
        for(int sum:temp)
        {
            money_sum.insert(sum);
        }
    }
    money_sum.erase(0);
    cout<<money_sum.size()<<endl;
    for(int it:money_sum)
    {
        cout<<it<< " ";
    }
```

Answer to the question no-06

```
#include<bits/stdc++.h>
using namespace std;
#define ll long long
const int SIZE=20005;
list<int> adj_list[SIZE];
int color[SIZE];
enum {NOT_VISITED, BLACK, RED};
int main()
{
   int tc;
   ll sum = 0;
   cin>>tc;
   for(int t = 1; t <= tc; t++)
   {
      int n;
      cin>>n;

      memset(color, 0, sizeof color);
      for(int i = 0; i < SIZE; i++)
      {
         adj_list[i].clear();
      }
      for(int i = 0; i < n; i++)
      {
         int u,v;
         cin>>u>>v;
         adj_list[u].push_back(v);
         adj_list[v].push_back(u);
      }
      ll mx = 0;
      for(int i = 0; i < SIZE; i++)
      {
         if(!adj_list[i].empty()&&color[i] == NOT_VISITED)
         {
            int black = 0, red = 0;
            queue<int> q;
            q.push(i);
            color[i] = BLACK;
            black++;
            while(!q.empty())
            {
               int node = q.front();
               q.pop();
               for(auto it:adj_list[node])
               {
                  if(color[it] == NOT_VISITED)
                  {
                     q.push(it);
                     if(color[node] == BLACK)
                     {
                        color[it] = RED;
                        red++;
                     }
                     else
                     {
                        color[it] = BLACK;
                        black++;
                     }
                  }
               }
            }
            mx += max(red, black);
         }
      }

      printf("Case %d: %d\n", t, mx);
   }

   return 0;
}
```

<u>Answer to the question no-07</u>

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

```
using namespace std;
#define ll long long
unordered_map<ll,ll>dp;
ll n,h;
ll required_length(ll k)
{
   if(n==1)
   {
      return -1;
   }
   if(dp[k]!=0)
   {
      return dp[k];
   }
   string s=to_string(k);
   if(s.size()>=n)
   {
      return 0;
   }
   int x=0,y=0;
   for(int i=0;i<s.size();i++)
   {
      if(s[i]=='1'||s[i]=='0')
      {
         x++;
      }
   }
   if(x==s.size()&&x>1)
   {
      dp[k]=1e9;
      return dp[k];
   }
   ll ans=1e9;
   for(char i:s)
   {
      int x=i-'0';
      if(x>1)
```

```
        {
           ans=min(ans,1+required_length(x*k));
        }
    }
    dp[k]=(ans==1e9?-1:ans);
    return dp[k];
}
int main()
{
    cin>>n>>h;
    string s=to_string(h);
    int x=0;
    for(int i=0;i<s.size();i++)
    {
        if(s[i]=='1'||s[i]=='0')
        {
           x++;
        }
    }
    if(x==s.size()&&x>1)
    {
        cout<<-1;
        return 0;
    }
    cout<<required_length(h)<<endl;
}
```

Answer to the question no-08

```
#include<bits/stdc++.h>
using namespace std;
const int N=3003;
#define ll long long
unordered_map<string,string>dp;
int n,m;
string LCS(string s,string t)
{
    if(s.empty()||t.empty())
    {
        return "";
    }
    if(dp.find(s+"|"+t)!=dp.end())
    {
        return dp[s+"|"+t];
    }
    if(s[0]==t[0])
    {
        string ans=s[0]+LCS(s.substr(1),t.substr(1));
        dp[s+"|"+t]=ans;
        return ans;
    }
    else
    {
        string ans1=LCS(s.substr(1),t);
        string ans2=LCS(s,t.substr(1));
        string ans=ans1.size()>ans2.size()?ans1:ans2;
        dp[s+"|"+t]=ans;
        return ans;
    }
}
int main()
{
    string s,t;
    cin>>s>>t;
    n=s.size();
    m=t.size();
    string a=LCS(s,t);
    cout<<a<<endl;
}
```