Link:

https://www.careercup.com/question?id=6217124124033024 (Packets and Queue)

https://www.geeksforgeeks.org/samsung-competency-test-25-aug-19/ {sinkholes}

Samsung Software Competency Test (SWC) for Working professionals - GeeksforGeeks (protein, fat, carbohidrat)

Samsung Interview Experience | Set 40 (On Campus White Box Testing) - GeeksforGeeks (at most 'k' ominous numbers.)

Samsung Interview Experience | Set 32 (On-Campus) - GeeksforGeeks (Physical energy) Solution:https://github.com/kaushal02/interview-coding-problems/blob/master/energyDifference.cpp

Samsung Semiconductor Institute of Research (SSIR software) intern/FTE | Set-1 - GeeksforGeeks (fishing)

Samsung R&D Bangalore | Interview Experience (On Campus FTE) - GeeksforGeeks (cycle detection proper question)

Samsung R & D Noida Question September 2018 - GeeksforGeeks (Electric banner{largest two pipe})

There are N cars parked in ar | CareerCup (Gasoline and diesel)

https://github.com/rsenwar/Samsung-Interview-Problems
https://github.com/s-kachroo/SamsungPractice
https://github.com/realabbas/big-companies-interview-questions/blob/master/companies/samsung/samsung.md
https://www.geeksforgeeks.org/samsung-competency-test-25-aug-19/
https://github.com/twowaits/SDE-Interview-Questions/tree/master/Samsung?fbclid=IwAR0MqnFFY6qoLibYIuo_sOVnfJmhQX2ZOS-3HWGHFxWx59YRmdHPxAzpqJ0
https://pastebin.com/u/ann8497
Samsung Question - general - CodeChef Discuss
https://www.codingninjas.com/codestudio/problems/minimum-time-in-wormhole-network_630527
https://github.com/MohMaya/TargetSMSNG/blob/master/README.md
https://www.codingninjas.com/codestudio/problems/closest-leaf-to-given-node-in-binary-tree_983627
http://www.shafaetsplanet.com/?p=3721

You can download Big Java from here:
https://www.pdfdrive.com/big-java-early-objects-e185377725.html

Pipes:

```
#include<iostream>
#include<string.h>
#include<stdio.h>
#define MX 1005

using namespace std;

int n, m, frnt = - 1, rear = -1;
int vis[MX][MX], level[MX][MX], arr[MX][MX];

struct node
{
    bool up, down, left, right;
} pipes[MX][MX];

struct point
{
    int x, y;
} que[1000005];

void push(int a, int b)
{
    if(frnt==-1)
        frnt = 0;
    rear++;
    rear = rear % 1000005;

    que[rear].x = a;
    que[rear].y = b;
    //cout<<level[a][b]<<endl;
}

void pop()
{
    frnt++;
    frnt = frnt % 1000005;
}

bool isEmpty()
```

```
{
    if(frnt==-1 || rear < frnt)
        return 1;
    return 0;
}

void init()
{
    memset(arr, 0, sizeof(arr));
    memset(vis, 0, sizeof(vis));
    memset(level, 0, sizeof(level));
}

bool isValid(int x, int y)
{
    if(x < n && y < m && x >= 0 && y >= 0)
        return 1;
    return 0;
}

int bfs(int r, int c, int l)
{
    if(arr[r][c]==0)
        return 0;

    push(r, c);
    vis[r][c] = 1;
    level[r][c] = 1;

    int ans = 1;
    while(!isEmpty())
    {
        int p = que[frnt].x;
        int q = que[frnt].y;
        pop();

        if(level[p][q] < l)
        {
            /* Row Up */
            if(isValid(p-1, q) && vis[p-1][q]==0 && pipes[p-1][q].down && pipes[p][q].up)
            {
                vis[p-1][q] = 1;
                level[p-1][q] = level[p][q] + 1;
                ans++;
```

```
            push(p-1, q);
        }
        /* Row down */
        if(isValid(p+1, q) && vis[p+1][q]==0 && pipes[p+1][q].up && pipes[p][q].down)
        {
            vis[p+1][q] = 1;
            level[p+1][q] = level[p][q] + 1;
            ans++;
            push(p+1, q);
        }
        /* Row left */
        if(isValid(p, q-1) && vis[p][q-1]==0 && pipes[p][q-1].right && pipes[p][q].left)
        {
            vis[p][q-1] = 1;
            level[p][q-1] = level[p][q] + 1;
            ans++;
            push(p, q-1);
        }
        if(isValid(p, q+1) && vis[p][q+1]==0 && pipes[p][q+1].left && pipes[p][q].right)
        {
            vis[p][q+1] = 1;
            level[p][q+1] = level[p][q] + 1;
            ans++;
            push(p, q+1);
        }
      }
   }
   return ans;
}

int main()
{
   //freopen("input.txt", "r", stdin);

   int t;
   cin>>t;

   while(t--)
   {
      int r, c, l;
      cin>>n>>m>>r>>c>>l;


      init();
```

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

        if( arr[i][j] == 1 )
        {
            pipes[i][j].left = true;
            pipes[i][j].right = true;
            pipes[i][j].up = true;
            pipes[i][j].down = true;
        }
        else if( arr[i][j] == 2 )
        {
            pipes[i][j].left = false;
            pipes[i][j].right = false;
            pipes[i][j].up = true;
            pipes[i][j].down = true;
        }
        else if( arr[i][j] == 3 )
        {
            pipes[i][j].left = true;
            pipes[i][j].right = true;
            pipes[i][j].up = false;
            pipes[i][j].down = false;
        }
        else if( arr[i][j] == 4 )
        {
            pipes[i][j].left = false;
            pipes[i][j].right = true;
            pipes[i][j].up = true;
            pipes[i][j].down = false;
        }
        else if( arr[i][j] == 5 )
        {
            pipes[i][j].left = false;
            pipes[i][j].right = true;
            pipes[i][j].up = false;
            pipes[i][j].down = true;
        }
        else if( arr[i][j] == 6 )
        {
            pipes[i][j].left = true;
            pipes[i][j].right = false;
```

```
                pipes[i][j].up = false;
                pipes[i][j].down = true;
            }
            else if( arr[i][j] == 7 )
            {
                pipes[i][j].left = true;
                pipes[i][j].right = false;
                pipes[i][j].up = true;
                pipes[i][j].down = false;
            }
            else
            {
                pipes[i][j].left = false;
                pipes[i][j].right = false;
                pipes[i][j].up = false;
                pipes[i][j].down = false;
            }
        }
    }

    int ans = bfs(r, c, l);
    cout<<ans<<"\n";
  }
  return 0;
}
```

Restroom:

```
#include<iostream>
#include<stdio.h>
using namespace std;

void solve(int n, int k, int arr[])
{
  for(int i=0; i<k; i++)
  {
    int l = 0, cnt = 0, ans = 0;
    for(int j=0; j<n; j++)
    {
      if(arr[j]==0)
        cnt++;
      else
        cnt = 0;
```

```
            if(cnt > ans)
            {
                ans = cnt;
                l = j;
            }
        }
        int f = l - ans + 1;
        int mid = (f+l)/2;
        arr[mid] = 1;
    }
}

int main()
{
    freopen("input.txt","r",stdin);
    int t;
    cin>>t;

    while(t--)
    {
        int n, k;
        cin>>n>>k;

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

        solve(n, k, arr);
        for(int i=0; i<n; i++)
            cout<<arr[i];
        cout<<"\n";
    }
    return 0;
}

/*
It is a well-researched fact that men in a restroom generally prefer to maximize their distance from already occupied stalls, by occupying the middle of the longest sequence of unoccupied places

10
10 1
10 2
```

10 3
10 4
10 5
10 6
10 7
10 8
10 9
10 10
output is the following

0000100000
0000100100
0100100100
0100100110
0100110110
0110110110
0110110111
0110111111
0111111111
1111111111

VERIFIED AND TESTED
*/

Spaceship Game:

```
#include<iostream>
#define MX 1005
using namespace std;

int arr[MX][6];
void bomb(int id, int n, int a[MX][6])
{
   for(int i=0; i<n; i++)
   {
      if(i >= id && i < id+5)
      {
         for(int j=0; j<5; j++)
         {
            if(arr[i][j]==2)
               a[i][j] = 0;
            else
```

```
                a[i][j] = arr[i][j];
            }
        }
        else
        {
            for(int j=0; j<5; j++)
            {
                a[i][j] = arr[i][j];
            }
        }
    }
}

int solve(int n, int a[MX][6])
{
    int p[] = {-1, -1, 0, -1, -1};
    int c[] = {-1, -1, -1, -1, -1};

    int ans = 0;
    for(int i=n-1; i>=0; i--)
    {
        for(int j=0; j<5; j++)
        {
            if(a[i][j]==2)
            {
                c[j] = -1;
                continue;
            }
            //cout<<a[i][j]<<endl;
            for(int k = max(0, j-1); k <= min(j+1, 4); k++)
            {
                if(p[k] != -1)
                {
                    c[j] = max(c[j], p[k] + a[i][j]);
                    //if(a[i][j]==1)
                       // cout<<i<<j<<" "<<c[j]<<endl;
                }
            }
            ans = max(ans, c[j]);
        }
        for(int l=0; l<5; l++)
            p[l] = c[l];
    }
    return ans;
```

```
}

int main()
{
    int t, cas = 0;
    cin>>t;

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

        for(int i=0; i<n; i++)
            for(int j=0; j<5; j++)
                cin>>arr[i][j];

        int a[n][6], ans = 0;
        if(n > 4)
        {
            for(int i=0; i<n-4; i++)
            {
                bomb(i, n, a);
                ans = max(ans, solve(n, a));
            }
        }
        else
        {
            for(int i=0; i<n; i++)
                for(int j=0; j<5; j++)
                    a[i][j] = arr[i][j];
            ans = max(ans, solve(n, a));
        }

        if(ans)
            cout<<"#"<< ++cas <<" "<<ans<<"\n";
        else
            cout<<"#"<< ++cas <<" 0\n";
    }
    return 0;
}



/*
```

4
7
1 2 0 0 1
2 0 0 1 0
0 1 2 0 1
1 0 0 2 1
0 2 1 0 1
0 1 2 2 2
1 0 1 1 0
5
1 1 0 0 0
1 2 2 2 1
1 1 2 2 1
2 2 2 1 2
2 2 0 2 0
6
2 2 2 2 2
0 0 0 0 0
0 0 2 0 0
2 0 0 0 2
0 0 0 0 0
1 2 2 2 1
12
2 2 2 2 2
1 1 0 1 1
0 1 0 1 0
1 0 1 0 1
2 2 0 0 2
1 1 0 0 1
2 2 2 2 2
1 1 0 1 0
0 1 0 1 0
1 0 1 0 1
2 2 2 2 2
2 2 0 1 1

Answers
6
3
-1
8

*/

```
// spaceship bomb
#include<bits/stdc++.h>
using namespace std;
#define INT_MIN -1000007
int ans = INT_MIN;

void solve(int board[][5], int i, int j, bool bombUsed, int rowValid, int coins){
	if(i<0 || j<0 || j>=5){
	ans = max(ans, coins);
	return;
	}
	if(board[i][j] == 1 || board[i][j] == 0){
	if(board[i][j] ==1){
	coins++;
	}
	if(bombUsed){
	rowValid--;
	}
	solve(board, i-1, j-1, bombUsed, rowValid, coins);
	solve(board, i-1, j, bombUsed, rowValid, coins);
	solve(board, i-1, j+1, bombUsed, rowValid, coins);
	}
	else if(board[i][j] ==2){
	if(bombUsed && rowValid <= 0){
	ans = max(ans, coins);
	return;
	}
	else if(bombUsed && rowValid > 0){
	rowValid--;
	solve(board, i-1, j-1, bombUsed, rowValid, coins);
	solve(board, i-1, j, bombUsed, rowValid, coins);
	solve(board, i-1, j+1, bombUsed, rowValid, coins);
	}
	else{
	bombUsed = true;
	rowValid = 4;
	solve(board, i-1, j-1, bombUsed, rowValid, coins);
	solve(board, i-1, j, bombUsed, rowValid, coins);
```

```
        solve(board, i-1, j+1, bombUsed, rowValid, coins);
        }
        }
}

int main(){
        int n;
        cin>>n;
        int board[n][5];
        for(int i =0; i<n; i++){
        for(int j = 0; j<5; j++){
        cin>>board[i][j];
        }
        }
        solve(board, n-1, 1, false, 0, 0);
        solve(board, n-1, 2, false, 0, 0);
        solve(board, n-1, 3, false, 0, 0);
        cout<<ans<<endl;
        return 0;
}
```

Research Team:

```
#include<iostream>
#include<string.h>
#include<stdio.h>
#define MX 1005
using namespace std;

int row, col, element;
int frnt = 0, rear = 0;
int vis[MX][MX], adj[MX][MX], level[MX][MX];

int dx[] = {0, 1, 0, -1};
int dy[] = {1, 0, -1, 0};

struct point
{
        int x, y;
} que[MX], loc[MX];
```

```
void init()
{
   frnt = rear = 0;
   memset(vis, 0, sizeof(vis));
}

void push(int a, int b)
{
        que[rear].x = a;
        que[rear].y = b;
        rear++;
}

void pop()f
{
        frnt++;
}

bool isEmpty()
{
        return (frnt == rear);
}

bool isValid(int x, int y)
{
        if(x>=0 && x<row && y>=0 && y<col)
        return 1;
        return 0;
}

int bfs(int sx, int sy, int vx, int vy)
{
        push(sx, sy);
        vis[sx][sy] = 1;
        level[sx][sy] = 0;

        while(!isEmpty())
        {
        int xx = que[frnt].x;
        int yy = que[frnt].y;
        pop();

        if(xx==vx && yy==vy)
        return level[xx][yy];
```

```
        for(int i=0; i<4; i++)
        {
        int ux = xx + dx[i];
        int uy = yy + dy[i];

        if(vis[ux][uy]==0 && isValid(ux, uy) && adj[ux][uy]==1)
        {
                vis[ux][uy] = 1;
                push(ux, uy);
                level[ux][uy] = level[xx][yy] + 1;
        }
        }
        }
        return -1;
}


int main()
{
   freopen("input.txt","r",stdin);
        int t;
        cin>>t;

        while(t--)
        {
        cin>>row>>element;
        col = row;

        for(int i=0; i<element; i++)
        {
        cin>>loc[i].x>>loc[i].y;
        }

        for(int i=0; i<row; i++)
        {
        for(int j=0; j<col; j++)
        {
                cin>>adj[i][j];
        }
        }

        int mx, ans = 100000;
        for(int i=0; i<row; i++)
```

```
        {
        for(int j=0; j<col; j++)
        {
                if(adj[i][j]==1)
                {
                   mx = 0;
                   for(int k=0; k<element; k++)
                {
                   init();
                mx = max(mx, bfs(i, j, loc[k].x-1, loc[k].y-1));
                //cout<<mx<<endl;
                }
                ans = min(ans, mx);
                }
        }
        }
        cout<<ans<<"\n";
        }
        return 0;
}

/*
remember the indexing
5
5 2
4 3
3 4
1 1 0 0 0
1 1 0 0 0
1 1 1 1 1
1 1 1 0 1
1 1 1 1 1
8 2
5 6
6 4
1 1 1 1 1 1 0 0
1 1 1 1 1 1 1 0
1 1 0 1 0 1 1 0
1 1 1 1 0 1 1 0
1 1 1 1 1 1 1 0
1 1 1 1 1 1 1 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
10 3
```

8 2
5 3
7 1
0 0 0 1 1 1 1 1 1 0
1 1 1 1 1 1 1 1 1 0
1 0 0 1 0 0 0 0 1 0
1 1 1 1 1 1 1 1 1 1
1 1 1 1 0 1 0 0 1 1
1 1 1 1 0 1 0 0 1 1
1 1 1 1 0 1 0 0 1 1
1 1 1 1 1 1 1 1 1 1
1 1 1 0 0 1 0 0 1 1
1 1 1 1 1 1 1 1 1 1
15 4
11 15
15 9
1 2
14 3
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 0 1 1 1 1 1 1 1 1 1 1 1 0 1
1 0 1 0 0 0 1 0 0 0 0 1 1 0 1
1 0 1 0 0 0 1 0 0 0 0 1 1 0 1
1 0 1 1 1 1 1 1 1 1 1 1 1 1 1
1 0 1 0 0 0 1 0 0 0 0 1 1 0 1
1 0 1 0 0 0 1 1 1 1 1 1 1 1 1
1 0 1 0 0 0 1 0 0 0 0 1 1 0 1
1 0 1 0 0 0 1 0 0 0 0 1 1 0 1
1 0 1 0 0 0 1 0 0 0 0 1 1 0 1
1 0 1 0 0 0 1 0 0 0 0 1 1 0 1
1 0 1 0 0 0 1 0 0 0 0 1 1 0 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
0 0 1 0 0 0 1 1 1 1 1 1 1 0 1
0 0 1 1 1 1 1 1 1 1 1 1 1 1 1
20 4
13 6
20 4
1 2
17 16
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0
1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0
1 0 1 0 0 0 0 0 0 0 1 0 0 1 1 0 0 0 0 0
1 0 1 0 0 0 0 0 0 0 1 0 0 1 1 0 0 0 0 0
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0
1 0 1 0 0 0 0 0 0 0 1 0 0 1 1 1 0 0 0 0

1 0 1 0 0 0 0 0 0 0 1 0 0 1 1 1 0 0 0 0
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 0 1 0 0 0 0 0 0 0 1 0 0 1 1 1 0 0 1 1
1 0 1 0 0 0 0 0 0 0 1 0 0 1 1 1 0 0 1 1
1 0 1 0 0 0 0 0 0 0 1 0 0 1 1 1 0 0 1 1
1 0 1 0 0 0 0 0 0 0 1 0 0 1 1 1 0 0 1 1
1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1
1 0 1 0 0 0 0 0 0 0 1 0 0 0 1 1 0 0 1 1
1 0 1 0 0 0 0 0 0 0 1 0 0 0 1 1 0 0 1 1
1 0 1 0 0 0 0 0 0 0 1 0 0 0 1 1 0 0 1 1
1 0 1 0 0 0 0 0 0 0 1 0 0 0 1 1 0 0 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0

Output
#1 1
#2 2
#3 2
#4 12
#5 15

*/

Days of Week:

```
#include<iostream>
#include<stdio.h>

using namespace std;

string week[] = { "Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday",
"Saturday" };

int dayOfWeek(int y, int m, int d)
{
	int t[] = { 12, 11, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };

	for (int i = 0; i < 12; i++)
```

```
        {
                int ans = t[i] * 2.6 - 0.2;
                t[i] = ans % 7;
        }

        int i = (y + y / 4 - y / 100 + y / 400 + t[m-1] + d) % 7;
        return i;
}

int main()
{
        int day, month, year;
        cin >> day >> month >> year;

        int ans = dayOfWeek(year, month, day);
        cout << week[ans] << "\n";
        return 0;
}
```

Aggressive Cows:

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

int binarySearch(int a[], int n, int c)
{
   int low = 0, high = a[n-1], best = 0;
   while(low <= high)
   {
      int mid = (low+high+1)/2;
      int cnt = 1, pre = 0;

      for(int i=1; i<n && cnt < c; i++)
      {
         if(a[i]-a[pre] >= mid)
            pre = i, cnt++;
      }

      if(c <= cnt)
```

```
        {
            best = mid;
            low = mid + 1;
        }
        else
            high = mid - 1;
    }
    return best;
}

int main()
{
    //freopen("input.txt","r",stdin);
    int t;
    cin>>t;

    while(t--)
    {
        int n, c;
        cin>>n>>c;

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

        sort(a, a+n);
        int ans = binarySearch(a, n, c);
        cout<<ans<<"\n";
    }
    return 0;
}
```

Graph Coloring:

```
#include<iostream>
#include<stdio.h>
#include<string.h>
#define MX 205

int adj[MX][MX], vis[MX], leb[MX];
int inp_arr[MX];
```

```
int Rear, Front;

void init()
{
   memset(vis, 0, sizeof(vis));
   memset(leb, 0, sizeof(vis));
   memset(inp_arr, 0, sizeof(vis));
   Rear = -1, Front = -1;
}

void push(int n)
{
   if (Front == - 1)
      Front = 0;
   Rear = Rear + 1;
   inp_arr[Rear] = n;
}

int pop()
{
   return inp_arr[Front++];
}

bool isempty()
{
   if (Front == - 1 || Front > Rear)
      return 1;
   else
      return 0;
}

int bfs(int n)
{
   vis[0] = 1;
   leb[0] = 1;
   push(0);

   while(!isempty())
   {
      int u = pop();
      //printf("%d\n",u);
      for(int i=0; i<n; i++)
      {
         if(vis[i]==0 && adj[u][i]==1)
```

```
            {
                vis[i] = 1;
                leb[i] = 1 - leb[u];
                push(i);
            }
            else if(vis[i]==1 && adj[u][i]==1)
                if(leb[i]==leb[u])
                    return 0;
        }
    }
    return 1;
}

int main()
{
    int n, m;
    while(scanf("%d",&n))
    {
        if(n==0)
            break;

        scanf("%d",&m);
        for(int i=0; i<m; i++)
        {
            int a, b;
            scanf("%d%d",&a,&b);
            adj[a][b] = 1;
            adj[b][a] = 1;
        }

        init();
        if(bfs(n))
            printf("BICOLORABLE.\n");
        else
            printf("NOT BICOLORABLE.\n");

        for(int i=0; i<=n; i++)
        {
            memset(adj[i], 0, sizeof(adj[i]));
        }
    }
    return 0;
}
```

Frog Jump:

```
#include<iostream>
#include<string.h>
#define MX 105
using namespace std;

int row, col;
int frnt = 0, rear = 0;
int vis[MX][MX], adj[MX][MX], level[MX][MX];

int dx[] = {0, 1, 0, -1};
int dy[] = {1, 0, -1, 0};

struct point
{
	int x, y;
} que[MX];

void init()
{
   frnt = rear = 0;
   memset(vis, 0, sizeof(vis));
}

void push(int a, int b)
{
	que[rear].x = a;
	que[rear].y = b;
	rear++;
}

void pop()
{
	frnt++;
}

bool isEmpty()
{
	return (frnt == rear);
}

bool isValid(int x, int y)
{
```

```
        if(x>=0 && x<row && y>=0 && y<col)
        return 1;
        return 0;
}

int bfs(int sx, int sy, int vx, int vy)
{
        push(sx, sy);
        vis[sx][sy] = 1;
        level[sx][sy] = 0;

        while(!isEmpty())
        {
        int xx = que[frnt].x;
        int yy = que[frnt].y;
        pop();

        for(int i=0; i<4; i++)
        {
        int ux = xx + dx[i];
        int uy = yy + dy[i];

        if(vis[ux][uy]==0 && isValid(ux, uy) && adj[ux][uy]==1)
        {
                vis[ux][uy] = 1;
                push(ux, uy);

                if(uy==yy)
            level[ux][uy] = level[xx][yy] + 1;
           else
            level[ux][uy] = level[xx][yy];

                if(ux==vx && uy==vy)
            return level[ux][uy];
        }
        }
        }
        return -1;
}

int main()
{
   cin>>row>>col;
   for(int i=0; i<row; i++)
```

```
        for(int j=0; j<col; j++)
            cin>>adj[i][j];

    int sx, sy, dx, dy;
    cin>>sx>>sy>>dx>>dy;

    int ans = bfs(sx, sy, dx, dy);
    cout<<ans<<endl;
    return 0;
}

/*
5 5
1 0 1 1 1
1 1 0 1 1
1 0 1 1 1
1 1 1 0 1
1 1 1 1 1
1 1 4 4
Output: 3 */
```

Cycle Detection:

```
#include<iostream>
#include<vector>
#include<string.h>
#include<algorithm>
#define MX 15
using namespace std;

vector<int>adj[MX], ans;
int vis[MX], par[MX], mx = 999999;

void dfs(int u, int p)
{
    if(vis[u]==1)
    {
        int sum = u;
        vector<int> cycle;
        cycle.push_back(u);

        for(int v = p; v != u; v = par[v])
```

```
        {
            //cout<<v<<endl;
            cycle.push_back(v);
            sum += v;
        }
        if(sum < mx)
        {
            ans = cycle;
            mx = sum;
        }
        return;
    }

    vis[u] = 1;
    par[u] = p;

    for(int i=0; i<adj[u].size(); i++)
    {
        int v = adj[u][i];
        dfs(v, u);
    }
    vis[u] = 0;
}

void findCycle(int n)
{
    memset(vis, 0, sizeof(vis));
    memset(par, -1, sizeof(par));

    for(int i=1; i<=n; i++)
    {
        if(!vis[i])
            dfs(i, -1);
    }

    sort(ans.begin(), ans.end());
    for(int i=0; i<ans.size(); i++)
        cout<<ans[i]<<" ";
    cout<<"\n";
}

int main()
{
    int n, m;
```

```
    cin>>n>>m;

    for(int i=0; i<m; i++)
    {
        int a, b;
        cin>>a>>b;
        adj[a].push_back(b);
    }

    findCycle(n);
    return 0;
}
```

MR Kim:

```
#include <iostream>
#include<stdlib.h>
#define MX 25
using namespace std;

struct point {
    int x, y;
}adj[MX];

int distance(point a, point b)
{
    return abs(a.x - b.x) + abs(a.y - b.y);
}

void Swap(point &a, point &b)
{
    point t = a;
    a = b;
    b = t;
}

int totalDistance(int n)
{
    int sum = 0;
```

```
    for (int i = 1; i < n; i++)
    {
        sum += distance(adj[i - 1], adj[i]);
    }
    return sum;
}

void permutation(point adj[], int l, int r, point s, point h, int &ans)
{
    if (l == r)
    {
        int tmp = distance(s, adj[0]) + totalDistance(r+1) + distance(h, adj[r]);
        //cout<<tmp<<endl;
        ans = min(ans, tmp);
        return;
    }
    else
    {
        for (int i = l; i <= r; i++)
        {
            Swap(adj[l], adj[i]);
            permutation(adj, l + 1, r, s, h, ans);
            Swap(adj[l], adj[i]);
        }
    }
}

int main()
{
    //freopen("input.txt","r", stdin);
    int t;
    cin >> t;

    while (t--)
    {
        int q, a, b;
        cin >> q;

        point s, h;
        cin >> s.x >> s.y >> h.x >> h.y;
```

```
        for (int i = 0; i < q; i++)
        {
            cin >> a >> b;
            adj[i].x = a;
            adj[i].y = b;
        }

        int ans = distance(s, adj[0]) + totalDistance(q) + distance(h, adj[q - 1]);
        permutation(adj, 0, q-1, s, h, ans);
        cout <<"The shortest path has length "<< ans << endl;
    }
    return 0;
}
```

Wormholes:

```
.#include<iostream>
#include <stdlib.h>
#include <stdio.h>
#define MX 405
using namespace std;

int vis[MX] = {0}, dist[MX];
struct point
{
        int sx, sy, dx, dy, time;
} holes[MX];

int distance(int x1, int y1, int x2, int y2)
{
        return abs(x1 - x2) + abs(y1 - y2);
}

int minDistance(int V)
{
        int min = INT_MAX, min_index;

        for (int v = 0; v < V; v++)
        if (vis[v] == false && dist[v] <= min)
```

```
        min = dist[v], min_index = v;
        return min_index;
}


int solve(int n)
{
        for(int i=0; i<n; i++)
        {
        int u = minDistance(n);
        vis[u] = 1;
       //cout<<u<<" "<<dist[u]<<endl;

        for(int v=0; v<n; v++)
        {

        int tmp1 = dist[u] + distance(holes[u].dx, holes[u].dy, holes[v].sx, holes[v].sy) +
holes[v].time;
        int tmp2 = dist[u] + distance(holes[u].dx, holes[u].dy, holes[v].dx, holes[v].dy);

        int tmp = min(tmp1, tmp2);

        if (!vis[v] && tmp < dist[v])
        {
                dist[v] = tmp;
                cout<<u<<v<<" "<<holes[v].sx<<" "<<holes[v].sy<<" "<<dist[v]<<endl;
        }
        }
        }
        return dist[n-1];
}

int main()
{
        freopen("input.txt","r",stdin);
        int sx, sy, dx, dy;
        cin>>sx>>sy>>dx>>dy;

        int worm;
        cin>>worm;
```

```
        int i=0;
        for(int j=0; j<worm; j++)
        {
        int a, b, c, d, e;
        cin>>a>>b>>c>>d>>e;

        holes[i].sx = a, holes[i].sy = b, holes[i].dx = c, holes[i].dy = d, holes[i].time = e;
        dist[i] = min(distance(sx, sy, holes[i].dx, holes[i].dy), distance(sx, sy, holes[i].sx,
holes[i].sy) + holes[i].time);
        i++;
        holes[i].sx = c, holes[i].sy = d, holes[i].dx = a, holes[i].dy = b, holes[i].time = e;
        dist[i] = min(distance(sx, sy, holes[i].dx, holes[i].dy), distance(sx, sy, holes[i].sx,
holes[i].sy) + holes[i].time);
        i++;
        //cout<<a<<" "<<b<<" "<<c<<" "<<" "<<d<<" "<<dist[i]<<endl;
        }

        holes[i].sx = dx, holes[i].sy = dy, holes[i].dx = dx, holes[i].dy = dy, holes[i].time =
0;
        dist[i] = distance(sx, sy, dx, dy);

        int ans = solve(i+1);
        cout<<ans<<endl;

        return 0;
}
```

Flip Columns:

```
#include<iostream>
#define MX 16
using namespace std;

int mx = 0;
int arr[MX][MX];

void flip(int n, int m, int j)
{
   for(int i=0; i<n; i++)
   {
```

```
        arr[i][j] = 1 - arr[i][j];
    }
}

void solve(int n, int m, int k)
{
    if(k == 0)
    {
        int ans = 0;
        for(int i=0; i<n; i++)
        {
            int cnt = 0;
            for(int j=0; j<m; j++)
            {
                if(arr[i][j]==1)
                    cnt++;
            }
            //cout<<cnt<<endl;
            if(cnt==m)
               ans++;
        }
        //cout<<ans<<endl;
        mx = max(mx, ans);
        return;
    }

    for(int i=0; i<m; i++)
    {
        flip(n, m, i);
        //cout<<k<<endl;
        solve(n, m, k-1);
        flip(n, m, i);
    }
}

int main()
{
    int n, m, k;
    cin>>n>>m>>k;
    for(int i=0; i<n; i++)
    {
        for(int j=0; j<m; j++)
        {
            cin>>arr[i][j];
```

```
        }
    }
    solve(n, m, k);
    cout<<mx<<endl;
    return 0;
}
```

Companies and oil mines:

```
#include<iostream>
#include<mem.h>
#define MX 101
#define INF 99999999
using namespace std;

int n, c, arr[2*MX];
int a[MX], dp[MX][MX][2];

int solve()
{
    int pre[n+1] = {0};
    for(int i=1; i<=n; i++)
        pre[i] = a[i] + pre[i-1];

    for(int j=1; j<=n; j++)
    {
        for(int k=1; k<=c && k<=j; k++)
        {
            if(k==1)
            {
                dp[j][k][0] = pre[j];
                dp[j][k][1] = pre[j];
                //cout<<pre[j]<<endl;
            }
            else
            {
                int tmp = INF;
                for(int l=j; l > k-1; l--)
                {
                    int mn = min(pre[j]-pre[l-1], min(dp[l-1][k-1][0], dp[l-1][k-1][1]));
                    int mx = max(pre[j]-pre[l-1], max(dp[l-1][k-1][0], dp[l-1][k-1][1]));
```

```
                if(tmp > mx-mn)
                {
                    tmp = mx-mn;
                    dp[j][k][0] = mn;
                    dp[j][k][1] = mx;
                }
            }
        }
    }
    return dp[n][c][1]-dp[n][c][0];
}

int main()
{
    int t;
    cin>>t;

    while(t--)
    {
        cin>>c>>n;
        for(int i=1; i<=n; i++)
            cin>>arr[i];

        if(c > n)
        {
            cout<<"-1\n";
            continue;
        }

        for(int i=1; i<=2*n; i++)
            arr[n+i] = arr[i];

        int ans = INF;
        for(int i=1; i<=2*n; i++)
        {
            int m = n + i -1, id = 0;
            for(int j=i; j<=m; j++)
            {
                a[++id] = arr[j];
            }
            memset(dp, -1, sizeof(dp));
            ans = min(ans, solve());
        }
```

```
        cout<<ans<<"\n";
    }
    return 0;
}

/*
2
2 4
6 13 10 2
2 4
6 10 13 2
*/
```

Maze and Jewels:

https://blog.csdn.net/broadCE/article/details/47959227

```
#include<iostream>
#include<string.h>
#include<stdio.h>
#define MX 105
using namespace std;

int row, col, mx = 0;
int ans[MX][MX], adj[MX][MX];;

int dx[] = {0, 1, 0, -1};
int dy[] = {1, 0, -1, 0};

bool isValid(int x, int y)
{
    if(x>=0 && x<row && y>=0 && y<col)
        return 1;
    return 0;
}

int dfs(int sx, int sy, int cnt)
{
    if(sx==row-1 && sy==row-1)
    {
        if(cnt > mx)
```

```
        {
            mx = cnt;
            for(int i=0; i<row; i++)
            {
                for(int j=0; j<col; j++)
                {
                    ans[i][j] = adj[i][j];
                }
            }
        }
    }

    for(int i=0; i<4; i++)
    {
        int ux = sx + dx[i];
        int uy = sy + dy[i];

        if(isValid(ux, uy) && (adj[ux][uy]==2 || adj[ux][uy]==0))
        {
            int check;
            if(adj[ux][uy]==2)
                check = 1;
            else
                check = 0;

            adj[ux][uy] = 3;
            dfs(ux, uy, cnt+check);

            if(check==1)
                adj[ux][uy] = 2;
            else
                adj[ux][uy] = 0;
        }
    }
}


int main()
{
    freopen("input.txt","r",stdin);
    int t;
    cin>>t;

    while(t--)
```

```
    {
        cin>>row;
        col = row;

        for(int i=0; i<row; i++)
        {
            for(int j=0; j<col; j++)
            {
                cin>>adj[i][j];
            }
        }

        mx = 0;
        adj[0][0] = 3;
        dfs(0,0,0);

        for(int i=0; i<row; i++)
        {
            for(int j=0; j<col; j++)
            {
                cout<<ans[i][j]<<" ";
            }
            cout<<"\n";
        }
        cout<<mx<<"\n";
    }
    return 0;
}

/*
remember the indexing
2
5
0 0 0 2 0
2 1 0 1 2
0 0 2 2 0
0 1 0 1 2
2 0 0 0 0
6
0 1 2 1 0 0
0 1 0 0 0 1
0 1 2 1 2 1
0 2 0 1 0 2
0 1 0 1 0 1
```

2 0 2 1 0 0

output:
Case #1

3 0 3 3 3
3 1 3 1 3
3 0 3 2 3
3 1 3 1 3
3 3 3 0 3
6

Case #2

3 1 2 1 0 0
3 1 3 3 3 1
3 1 3 1 3 1
3 2 3 1 3 2
3 1 3 1 3 1
3 3 3 1 3 3
4
*/

Mr. Lee Offices:

```
#include <iostream>
#include <climits>
#define EMPTY_VALUE -1
#define MAX_N 10
#define INF 1061109567
using namespace std;

int w[MAX_N][MAX_N], n;
int mem[MAX_N][1 << MAX_N];

int turnOn(int x, int pos) {
    return n | (1 << pos);
}

bool isOn(int x, int pos) {
    return (bool)(x & (1 << pos));
}
```

```
int f(int i, int mask) {
    if (mask == (1 << n) - 1) {
        return w[i][0];
    }

    if (mem[i][mask] != -1) {
        return mem[i][mask];
    }

    int ans = INF;
    for (int j = 0;j < n;j++) {
       // if (w[i][j] == 0) continue;

        if (isOn(mask, j) == 0) {
            int result = f(j, turnOn(mask, j)) + w[i][j];
            ans = min(ans, result);
        }
    }
    return mem[i][mask] = ans;
}

int main()
{
    int t;
    cin >> t;

    while (t--)
    {
        cin>>n;
        for (int i = 0; i < n; i++)
            for (int j = 0; j < n; j++)
                cin >> w[i][j];

        int ans = f(0, 1);
        cout << ans << "\n";
    }
    return 0;

}
```

Chessboard:

https://www.cnblogs.com/kingshow123/p/practicec2.html

```
#include<bits/stdc++.h>
#define pii pair<int, int>
#define MX 30
using namespace std;

int n, m;
int vis[MX][MX], level[MX][MX];

int dx[] = {-1, 1, -1, 1, -2, -2, 2, 2};
int dy[] = {-2, -2, 2, 2, -1, 1, -1, 1};

bool isValid(int x, int y)
{
    if(x>=0 && x<n && y>=0 && y<m)
        return 1;
    return 0;
}

int bfs(int sx, int sy, int ex, int ey)
{
    queue<pii>q;
    q.push({sx, sy});
    vis[sx][sy] = 1;
    level[sx][sy] = 0;

    while(!q.empty())
    {
        pii u = q.front();
        q.pop();

        int x = u.first;
        int y = u.second;

        for(int i=0; i<8; i++)
        {
            int vx = x + dx[i];
            int vy = y + dy[i];
```

```
            if(isValid(vx, vy) && !vis[vx][vy])
            {
                q.push({vx,vy});
                vis[vx][vy]=1;
                level[vx][vy]= level[x][y] + 1;

                if(vx==ex && vy==ey)
                    return level[vx][vy];
            }
        }
    }
    return -1;
}

int main()
{
    int t, cas = 0;
    cin>>t;

    while(t--)
    {
        int c, s, r, k;
        cin>>n>>m>>r>>c>>s>>k;

        memset(vis, 0, sizeof(vis));
        int ans = bfs(r, c, s, k);
        cout<<"Case #"<< ++cas <<"\n"<<ans<<"\n";
    }
    return 0;
}
```

Crow Pots:

```
#include <iostream>
#include<algorithm>

#define MX 1005
using namespace std;

int arr[MX], dp[MX][MX];
int solve(int n, int z)
{
```

```
    int ans = 99999999;
    for (int i = 1; i <= n; i++)
        dp[i][1] = arr[i] * (n - i + 1);

    for(int i=n; i>0; i--)
        for(int j=2; j<=z; j++)
            for(int k=i+1; k<=n; k++)
                dp[i][j] = min(dp[i][j], dp[k][j - 1] + (k - i) * arr[i]);

    for (int i = 1; i <= n; i++)
        ans = min(ans, dp[i][z]);
    return ans;
}

int main()
{
    int t;
    cin >> t;

    while (t--)
    {
        int n, k;
        cin >> n >> k;

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

        for (int i = 0; i < MX; i++)
            for (int j = 0; j < MX; j++)
                dp[i][j] = 9999999;

        sort(arr + 1, arr + n + 1);
        int ans = solve(n, k);
        cout << ans << "\n";
    }
    return 0;
}
```

Biochemical Laughing Bomb:

https://www.cnblogs.com/kingshow123/p/practicec1.html

```
#include<iostream>
#include<stdio.h>
#include<string.h>
#define MX 105
using namespace std;

int dx[] = { 0, 1, 0, -1 };
int dy[] = { 1, 0, -1, 0 };
int n, m, arr[MX][MX];

int frnt = -1, rear = -1;
int level[MX][MX];

struct point {
	int x, y;
}que[MX];

void push(int a, int b)
{
	if (frnt == -1)
		frnt = 0;
	rear++;

	que[rear].x = a;
	que[rear].y = b;
}

void pop()
{
	frnt++;
}

bool isEmpty()
{
	if (frnt == -1 || rear < frnt)
		return 1;
	return 0;
}

bool isValid(int x, int y)
{
	if (x < m && y < n && x >= 0 && y >= 0)
		return 1;
```

```
	else
		return 0;
}

int bfs(int sx, int sy)
{
	int ans = 0;
	push(sx, sy);
	arr[sx][sy] = 1;
	level[sx][sy] = 1;

	while (!isEmpty())
	{
		int xx = que[frnt].x;
		int yy = que[frnt].y;
		pop();

		for (int i = 0; i < 4; i++)
		{
			int ux = xx + dx[i];
			int uy = yy + dy[i];

			if (isValid(ux, uy) && arr[ux][uy]==1)
			{
				level[ux][uy] = level[xx][yy] + 1;
				arr[ux][uy] = 2;
				push(ux, uy);
				ans = level[ux][uy];
			}
		}
	}
	return ans;
}

int main()
{
	int t;
	cin >> t;

	while (t--)
	{
		cin >> n >> m;
		for (int i = 0; i < m; i++)
			for (int j = 0; j < n; j++)
```

```
                                cin >> arr[i][j];

                int x, y;
                cin >> x >> y;
                int ans = bfs(x-1, y-1);
                cout << ans << "\n";
        }
        return 0;
}
```

Balloons and Bullets:

```
#include <iostream>
#include<vector>
#include<string.h>
#define MX 15
using namespace std;

int dp[15][15];
int solve(int i, int j, vector<int> &a)
{
   if(i > j)
      return 0;

   if(dp[i][j] >= 0)
      return dp[i][j];

   int mx = -1;
   for(int k=i; k<=j; k++)
   {
      int sum = a[i-1] * a[j+1] + solve(i, k-1, a) + solve(k+1, j, a);
      dp[i][j] = mx = max(sum, mx);
   }
   return dp[i][j] = mx;
}

int maxCoins(vector<int>& nums)
{
   int n = nums.size();
   nums.push_back(1);
   nums.insert(nums.begin(), 1);

   memset(dp, -1, sizeof(dp));
```

```
    int ans = solve(1, n, nums);
    return ans;
}

int main()
{
    int n, a, mx = 0;
    cin>>n;

    vector<int> v;
    for(int i=0; i<n; i++)
    {
        cin>>a;
        v.push_back(a);
        mx = max(mx, a);
    }

    int ans = maxCoins(v) + mx - 1;
    cout<<ans<<"\n";
    return 0;
}
```

K Ominos Number:

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

int mp[11] = {0};

int solve(int num, int times)
{
    int cnt = 0;
    while (num)
    {
        int r = num % 10;
        if (mp[r])
            cnt++;
        num /= 10;
    }

    if (cnt >= times)
        return 1;
    return 0;
```

```
}

int main()
{
    int start, end;
    cin >> start >> end;

    int n, a;
    cin >> n;

    for (int i = 0; i < n; i++)
    {
        cin >> a;
        mp[a] = 1;
    }

    int times;
    cin >> times;

    int cnt = 0;
    for (int i = start; i <= end; i++)
    {
        if (solve(i, times))
            cnt++;
    }
    cout << cnt << "\n";
    return 0;
}
```

Fishery Gates:

```
#include<iostream>
#include<string.h>
#include<stdio.h>
#define MX 1005
#define INF 999999
using namespace std;

int pos[MX], val[MX], vis[MX];
int n, ans;

int posLeft(int pos)
{
    for(int i=pos-1; i>0; i--)
```

```
        if(vis[i]==0)
            return i;
    return INF;
}

int posRight(int pos)
{
    for(int i=pos; i<=n; i++)
        if(vis[i]==0)
            return i;
    return INF;
}

void solve(int x, int y, int z)
{
    int arr[] = {x, y, z}, sum = 0;
    for(int i=0; i<3; i++)
    {
        int id = arr[i];
        for(int j=1; j<=val[id]; j++)
        {
            int l = posLeft(pos[id]);
            int r = posRight(pos[id]);

            int ll, rr;
            if(l >= INF)
                ll = l;
            else
                ll = pos[id] - l + 1;

            if(r >= INF)
                rr = r;
            else
                rr = r - pos[id] + 1;
            //cout<<pos[id]<<" "<<ll<< " " <<rr<<endl;
            if(ll < rr)
            {
                vis[l] = 1;
                sum += ll;
            }
            else
            {
                vis[r] = 1;
                sum += rr;
```

```
            }
        }
    }
    ans = min(ans, sum);
}

int main()
{
    //freopen("input.c", "r", stdin);
    int t=1;
   // cin>>t;

    while(t--)
    {
        cin>>n;
        for(int i=1; i<4; i++)
            cin>>pos[i];
        for(int i=1; i<4; i++)
            cin>>val[i];

        ans = INF;
        memset(vis, 0, sizeof(vis));
        solve(1, 2, 3);
        memset(vis, 0, sizeof(vis));
        solve(1, 3, 2);
        memset(vis, 0, sizeof(vis));
        solve(2, 1, 3);
        memset(vis, 0, sizeof(vis));
        solve(2, 3, 1);
        memset(vis, 0, sizeof(vis));
        solve(3, 1, 2);
        memset(vis, 0, sizeof(vis));
        solve(3, 2, 1);
        cout<<ans<<"\n";
    }
    return 0;
}



/*
inputs
5
10
```

4 5
6 2
10 2
10
8 5
9 1
10 2
24
15 3
20 4
23 7
39
17 8
30 5
31 9
60
57 12
31 19
38 16
outputs
18
25
57
86
339
*/

Electric banner largest two pipe:

https://www.geeksforgeeks.org/samsung-r-d-noida-question-september-2018/

```
#include<iostream>
#include<string.h>
#define MX 105
using namespace std;

int n, arr[MX];
int dp[999][999][MX];

int solve(int p1, int p2, int i, int mx_p)
{
    if(p1 > mx_p || p2 > mx_p || i > n)
        return -1;
```

```
    if(dp[p1][p2][i] != -1)
        return dp[p1][p2][i];

    if(p1==p2)
        dp[p1][p2][i] = max(dp[p1][p2][i], p1);

    dp[p1][p2][i] = max(dp[p1][p2][i], solve(p1, p2, i+1, mx_p));
    dp[p1][p2][i] = max(dp[p1][p2][i], solve(p1+arr[i], p2, i+1, mx_p));
    dp[p1][p2][i] = max(dp[p1][p2][i], solve(p1, p2+arr[i], i+1, mx_p));
    return dp[p1][p2][i];
}

int main()
{
    int t;
    cin>>t;

    while(t--)
    {
        cin>>n;

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

        memset(dp, -1, sizeof(dp));
        int ans = solve(0, 0, 0, sum/2);
        cout<<ans<<"\n";
    }
    return 0;
}
```

Gasoline and diesel:

```
#include<iostream>
#include<stdlib.h>
#include<string.h>
#define MX 1005
#define INF 9999999
```

```
using namespace std;

int car[MX], N, ans = INF;
bool vis[MX];

void solve(int id, int cost, int cnt, int type, int rem, int ins)
{
   if(cnt==N)
   {
      ans = min(ans, cost);
      return;
   }
   if(rem <= 0)
      return;

   if(ins==0 && type==1)
   {
      int dist;
      for(int i=1; i<=N; i++)
      {
         if(!vis[i] && car[i]==1)
         {
            vis[i] = 1;
            dist = abs(id-i);
            solve(i, cost+dist, cnt+1, type, rem-1, 0);
            solve(i, cost+dist, cnt+1, type, 2, 1);
            solve(i, cost+dist, cnt+1, type, 2, 2);
            vis[i] = 0;
         }
      }
   }

   if(ins==0 && type==2)
   {
      int dist;
      for(int i=1; i<=N; i++)
      {
         if(!vis[i] && car[i]==2)
         {
            vis[i] = 1;
            dist = abs(id-i);
            solve(i, cost+dist, cnt+1, type, rem-1, 0);
            solve(i, cost+dist, cnt+1, type, 2, 1);
            solve(i, cost+dist, cnt+1, type, 2, 2);
```

```
                vis[i] = 0;
            }
        }
    }
    if(ins==1)
        solve(0, cost+id, cnt, 1, 2, 0);
    if(ins==2)
        solve(N+1, cost+(N+1-id), cnt, 2, 2, 0);
}

int main()
{
    int t, cas = 0;
    cin>>t;

    while(t--)
    {
        cin>>N;
        for(int i=1; i<=N; i++)
            cin>>car[i];

        ans = INF;
        memset(vis, 0, sizeof(vis));
        solve(0, 0, 0, 1, 2, 0);

        if(ans==INF)
        {
            memset(vis, 0, sizeof(vis));
            solve(N+1, N+1, 0, 2, 2, 0);
        }
        cout<<"# "<< ++cas <<" "<<ans<<endl;
    }
    return 0;
}
/*
2
5
1 2 1 2 1
5
2 1 1 2 1
*/
```

Rock Climbing:

```
#include<iostream>
#include<string.h>
using namespace std;

int n, m, f = 0;
int a[15][15], v[15][15];

bool isValid(int x, int y)
{
   if(x>=0 && x<n && y>=0 && y<m && (a[x][y]==1 || a[x][y]==3) && v[x][y]==0)
      return 1;
   return 0;
}

void dfs(int i, int j, int l)
{
   if(i<0 || j<0 || i>=n || j>=m )
      return;

   if(v[i][j])
      return;

   if(a[i][j]==3)
   {
      f = 1;
      return;
   }

   v[i][j] = 1;
   if(isValid(i, j+1))
      dfs(i, j+1, l);

   if(isValid(i, j-1))
      dfs(i, j-1, l);

   for(int h=1; h<=l; h++)
      if(isValid(i-h, j))
         dfs(i-h, j, l);

   for(int h=1; h<=l; h++)
      if(isValid(i+h, j))
         dfs(i+h, j, l);
```

```
}

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

    for(int l=0; l<n; l++)
    {
        memset(v, 0, sizeof(v));
        f = 0;
        dfs(n-1, 0, l);
        if(f)
        {
            cout<<l<<endl;
            break;
        }
    }
    return 0;
}
```

Unique BST:

```
#include<iostream>
#include<string.h>
#define MX 1005
using namespace std;

int dp[MX][MX];
int solve(int n, int k)
{
    if(k==0 || n==k)
        return 1;

    if(dp[n][k] >= 0)
        return dp[n][k];

    return dp[n][k] = solve(n-1, k-1) + solve(n-1, k);
}

int main()
{
```

```
    int n;
    cin>>n;

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

    memset(dp, -1, sizeof(dp));
    int ans = solve(2*n, n)/(n+1);
    cout<<ans<<endl;
}
```

Physical Energy

```
#include<iostream>
#include<string.h>
#include<stdio.h>
#define MX 105
#define INF 1000000

using namespace std;

int cost[MX], time[MX];
int dp[4040][1010][5];

int solve(int h, int d, int n)
{
    if(h < 0 || n==0)
        return INF;

    if(d==0)
        return 0;

    if (dp[h][d][n] != -1)
        return dp[h][d][n];

    return dp[h][d][n] = min(solve(h, d, n-1), time[n-1] + solve(h-cost[n-1], d-1, n));
}

int main()
{
    freopen("input.txt","r",stdin);
    int n, h, d;
    cin>>n>>h>>d;
```

```
    for(int i=0; i<n; i++)
    {
        int c, t;
        cin>>c>>t;
        cost[i] = c, time[i] = t;
    }

    memset(dp, -1, sizeof(dp));
    int ans = solve(h, d, n);
    cout<<ans<<"\n";
    return 0;
}

/*
you are a marathoner. you want to break previous record.
you have given 5 speed and corresponding stamina loss (ex: Speed1: 3 min 20 sec per unit, stamina loss1: 15).
it is possible switch between speed after one unit interval.
your total stamina H(1<=H<=160), you have to run distance D(1<=D<=40).
find minimum time to finish marathon within your stamina(you can't run with stamina<=0).*/
```

Sum of Nodes in Kth Level

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

int solve(string s, int k)
{
    int l = 0, ans = 0;
    for(int i=0; i<s.size(); i++)
    {
        if(s[i]=='(')
            l++;
        else if(s[i]==')')
            l--;
        else if(l==k+1)
        {
            int temp = 0;
            int bit = (s[i] - '0');
            while(bit>=0 && bit<=9)
            {
```

```
            temp = temp * 10 + bit;
            i++;
            bit = (s[i] - '0');
        }
        //cout<<temp<<endl;
        ans += temp;
        i--;
      }
    }
    return ans;
}

int main()
{
    int k;
    cin>>k;

    string s;
    cin>>s;

    int ans = solve(s, k);
    cout<<ans<<"\n";
    return 0;
}
```

Size of the largest subtree BST:

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

class node
{
public:
    int data;
    node* left;
    node* right;

    node(int data)
    {
        this->data = data;
        this->left = NULL;
        this->right = NULL;
    }
};
```

```
int isBSTUtil(node* node, int min, int max)
{
   if (node==NULL)
      return 1;

   if (node->data < min || node->data > max)
      return 0;

   return
      isBSTUtil(node->left, min, node->data-1) && // Allow only distinct values
      isBSTUtil(node->right, node->data+1, max); // Allow only distinct values
}


int isBST(node* node)
{
   return(isBSTUtil(node, INT_MIN, INT_MAX));
}

int sizeOfTree(node* node)
{
   if (node == NULL)
      return 0;
   else
      return(sizeOfTree(node->left) + 1 + sizeOfTree(node->right));
}

int largestBST(struct node *root)
{
   if(root == NULL)
      return 0;

   if (isBST(root))
      return sizeOfTree(root);
   else
      return max(largestBST(root->left), largestBST(root->right));
}

node* insert(int data, node*root)
{
   if(root == NULL)
   {
      node *temp = new node(data);
```

```
        return temp;
    }

    if(data > root->data) root->right = insert(data, root->right);
    else if(data < root->data) root->left =  insert(data, root->left);
    return root;
}

int main()
{
    node *root = new node(4);
    root->left = new node(2);
    root->right = new node(5);
    root->left->left = new node(1);
    root->left->right = new node(3);

    root = insert(10, root);

    int ans = largestBST(root);
    cout<<ans<<"\n";
    return 0;
}
```