Dijkstra:

MY task is to find the minimum cost value to go from the topleft corner to the bottom-right corner of a given 2D - number maze of size N × M where 1 ≤ N, M ≤ 999 .

CODE:

```
#include<bits/stdc++.h>
using namespace std;
#define ll long long
ll edge[1005][1005],cost[1005][1005];
#define inf 0x3f3f3f3f
long long  dis[1005][1005];
int fx[]= {+1,-1,+0,+0};
int fy[]= {+0,+0,+1,-1};
ll row,col;

class node
{
public:
   long long  x,y,costt;
   bool operator < (const node & b)const
   {
      return costt > b.costt;
   }
};

long long  dijkstra(ll srcx,ll srcy,ll desx,ll desy)
{
   memset(cost,inf,sizeof(cost));
   priority_queue<node>pq;
   node u,v;
   u.x = srcx;
   u.y = srcy;
   u.costt = dis[srcx][srcy];
   pq.push(u);
   cost[srcx][srcy] = dis[srcx][srcy];
   while(!pq.empty())
   {
      u = pq.top();
      pq.pop();
      long long  xx = u.x;
      long long  yy = u.y;
      long long  cst = cost[xx][yy];
      for(long long  i=0; i<4; i++)
      {
///         v.city = edge[ct][i];
///          v.costt= cost[ct][i] + cst;
         v.x = xx + fx[i];
         v.y = yy + fy[i];
         if(v.x>=0 && v.x<row && v.y>=0 && v.y<col){
         v.costt = dis[v.x][v.y] + cst;
         if(v.costt<cost[v.x][v.y])
         {
            cost[v.x][v.y] = v.costt;
            pq.push(v);
         }
      }
      }
   }
   return cost[desx][desy];

}
int main()
{
   freopen("output.txt","wt",stdout);
   long long  t;
   scanf("%lld",&t);
   while(t--)
   {
      scanf("%lld %lld",&row,&col);
      for(ll i=0; i<row; i++)
         for(ll j=0; j<col; j++)
            scanf("%lld",&dis[i][j]);
      ll val = dijkstra(0,0,row-1,col-1);
      cout<<val<<endl;
   }
   return 0;
}
```