```
/*** Lightoj 1424- New Land:
You are given an r(row) x c(column) grid consists of only 0 or 1.
Now your task is to find the maximum rectangular area that consists only zeroes(0).
Input starts with an integer T (≤ 4), denoting the number of test cases.
Each case starts with a line containing two integers: r and c (1 ≤ r, c ≤ 2000).
Each of the next r lines contains c characters (either 0 or 1) denoting the kingdom.
For each case, print the maximum rectangular area that consists only zeroes (0).
Input:
1
5 5
10010
10001
00000
10000
11010
Output: 9
***/
#include<bits/stdc++.h>
using namespace std;
#define mx 2005
stack<int>st;
char ss[mx][mx];
int aa[mx][mx],LL[mx][mx],RR[mx][mx];
int main(){
    int tt; scanf("%d",&tt);
    for(int ks=1; ks<=tt; ks++){
        int rr,cc; scanf("%d%d",&rr,&cc);
        for(int i=1; i<=rr; i++)scanf("%s",ss[i]);

        for(int i=1; i<=rr; i++){
            for(int j=1; j<=cc; j++){
                if(ss[i][j-1]=='1')aa[i][j] = 0;
                else aa[i][j] = aa[i-1][j]+1;
            }
        }

        for(int i=1; i<=rr; i++){
            for(int j=1; j<=cc; j++){
                while(!st.empty() && aa[i][st.top()]>aa[i][j]){
                    int id = st.top(); st.pop();
                    RR[i][id] = j-1;
                }
                st.push(j);
            }
            while(!st.empty()){
                int id = st.top(); st.pop();
                RR[i][id] = cc;
            }

            for(int j=cc; j>=1; j--){
                while(!st.empty() && aa[i][st.top()]>aa[i][j]){
                    int id = st.top(); st.pop();
                    LL[i][id] = j+1;
                }
                st.push(j);
            }
            while(!st.empty()){
                int id = st.top(); st.pop();
                LL[i][id] = 1;
            }
        }

        int ans=0;
        for(int i=1; i<=rr; i++){
            for(int j=1; j<=cc; j++){
                int area = (RR[i][j]-LL[i][j]+1)*aa[i][j];
                ans = max(ans,area);
            }
        }
        printf("Case %d: %d\n",ks,ans);
    }
}
```

```cpp
#include<bits/stdc++.h>
using namespace std;
#define mx 2005
char ss[mx][mx];
int aa[mx][mx], tree[4*mx+5];
void init(int nd,int b,int e,int id){
    if(b==e){ tree[nd]=b; return; }

    int lf=nd*2, rg=nd*2+1, md=(b+e)/2;

    init(lf,b,md,id);
    init(rg,md+1,e,id);

    if(aa[id][tree[lf]]<=aa[id][tree[rg]]) tree[nd]=tree[lf];
    else tree[nd]=tree[rg];
}
int query(int nd,int b,int e,int x,int y,int id){
    if(x>e||y<b) return 0;
    if(b>=x&&e<=y) return tree[nd];

    int lf=nd*2, rg=nd*2+1, md=(b+e)/2;

    int m1=query(lf,b,md,x,y,id);
    int m2=query(rg,md+1,e,x,y,id);

    if(m1==0) return m2;
    if(m2==0) return m1;
    if(aa[id][m1]<=aa[id][m2]) return m1;
    else return m2;
}
int CalculateMaxArea(int b,int e,int n,int id){
    if(b<1||e<1||e<b) return 0;

    int maxx=0;
    int ps=query(1,1,n,b,e,id);
    int area=(e-b+1)*aa[id][ps];
    int area2=CalculateMaxArea(b,ps-1,n,id);
    int area3=CalculateMaxArea(ps+1,e,n,id);
    return maxx=max(maxx,max(area,max(area2,area3)));
}
int main(){
    int tt; scanf("%d",&tt);
    for(int ks=1; ks<=tt; ks++){
        int rr,cc; scanf("%d%d",&rr,&cc);
        for(int i=1; i<=rr; i++) scanf("%s",ss[i]);

        for(int i=1; i<=rr; i++){
            for(int j=1; j<=cc; j++){
                if(ss[i][j-1]=='1')aa[i][j] = 0;
                else aa[i][j] = aa[i-1][j]+1;
            }
        }

        int ans=0;
        for(int i=1; i<=rr; i++){
            init(1,1,cc,i);
            int area = CalculateMaxArea(1,cc,cc,i);
            ans = max(ans,area);
        }
        printf("Case %d: %d\n",ks,ans);
    }
    return 0;
}
```