A

将N和X转换为2进制形式，可以看出若X的最高位为k，则若N的第k位也为1，异或后该位变为0，不论后面的低位数字如何变，影响都小于该位，因此异或后的大小必定小于N。
因此判断统计N二进制的每一位为1的数值就能得到答案，注意判断最大值即可。

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

int main()
{
    ll N, R;
    while (~scanf("%lld %lld", &N, &R))
    {
        ll ans = 0;
        for (ll i = 0; i <= 62; i++)
        {
            if (N & (1LL << i))
            {
                ll l = 1LL << i, r = min(R, (1LL << (i + 1)) - 1);
                ans += max(0LL, r - l + 1);
            }
        }
        printf("%lld\n", ans);
    }
    return 0;
}
```

B

分别统计两行的前缀和。随后就可以O(1)的求出小P在某一列往下走时，小T可以取到的最大值，取它们的最小值即可。

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

const int maxn = 1e5 + 5;
int sum[2][maxn];

int main()
{
    int n, i, j, k;
    scanf("%d", &n);
    for (i = 0; i < 2; i++)
    {
        for (j = 1; j <= n; j++)
        {
            scanf("%d", &k);
            sum[i][j] = sum[i][j - 1] + k;
        }
```

```
    }

    int ans = 2e9;
    for (i = 1; i <= n; i++)
    {
        int res = max(sum[0][n] - sum[0][i], sum[1][i - 1]);
        ans = min(ans, res);
    }
    printf("%d\n", ans);
    return 0;
}
```

C

注意转义字符直接输出即可。

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

int main()
{
    printf("\"\"\"\n");
    printf("#include<stdio.h>\n\n");
    printf("int main()\n");
    printf("{\n");
    printf("    int a, b;\n");
    printf("    scanf(\"%%d %%d\", &a, &b);\n");
    printf("    printf(\"%%d\\n\", a + b);\n");
    printf("    return 0;\n");
    printf("}\n");
    printf("\"\"");
    return 0;
}
```

D

预处理出到1e18的阶乘，从大到小贪心组合即可。

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

int main()
{
    ll a[30], x;
    a[0] = 1;
    for (int i = 1; i <= 20; i++)
    {
        a[i] = a[i - 1] * i;
    }
    while (~scanf("%lld", &x))
    {
        int ans = 0, i = 20;
        while (x)
        {
```

```
            ans += x / a[i];
            x = x % a[i];
            i--;
        }
        printf("%d\n", ans);
    }
    return 0;
}
```

E

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

int main()
{
    int n, t;
    scanf("%d %d", &n, &t);
    while (t--)
    {
        int x, y;
        scanf("%d %d", &x, &y);
        if (n & (1 << (x - 1)))
        {
            n -= 1 << (x - 1);
            n += 1 << (x + y - 1);
        }
        printf("%d\n", n);
    }
    return 0;
}
```

F

求出中位数的最小值和最大值，中位数是可以从最小值到最大值线性增加的，即中位数种数为 max - min + 1

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

const int maxn = 1006;
int a[maxn],b[maxn];

int main()
{
    int n;
    scanf("%d",&n);
    for(int i=0;i<n;i++)
        scanf("%d %d",&a[i], &b[i]);
    sort(a, a+n);
    sort(b, b+n);
    printf("%d\n",b[n/2] - a[n/2] + 1);
    return 0;
}
```

G

```
#include <bits/stdc++.h>
using namespace std;
const int maxn = 2005;
char a[maxn];

int main()
{
    int n;
    scanf("%s %d", a, &n);
    int y = strlen(a), i;
    for (i = 1; i <= n; i++)
        a[i] = (i + 1 < y) ? a[i + 1] : '0';
    a[i] = i + 1 < y ? '.' : '\0';
    printf("%s\n", a);
    return 0;
}
```

H

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

int main()
{
    int n, m;
    scanf("%d %d",&n,&m);
    printf("%d\n", m - n%m);
    return 0;
}
```