蛙蛙跳跃

> 21351023 李永琦

题目描述

池塘中共有 n 片荷叶，蛙蛙初始在第 1 片荷叶，想要跳往第 n 片荷叶。当蛙蛙在第 i 片荷叶时，它能跳往第 j 片荷叶当且仅当 $j \oplus i < j$ 且 $j > i$（其中 $\oplus$ 表示按位异或）。请问蛙蛙有多少种不同的跳跃方式前往第 n 片荷叶。若不可能到达第 n 片荷叶，则认为方案数为 0。由于答案可能很大，请你对 998244353 取模。

我们认为两种跳跃路线 $a_1 \to a_2 \to \cdots \to a_n$ 与 $b_1 \to b_2 \to \cdots \to b_m$ 不同当且仅当 $n \neq m$ 或存在 i $(1 \le i \le \min(n, m))$ 使得 $a_i \neq b_i$。

输入

第一行为数据组数 T $(1 \le T \le 10^4)$。

接下来 T 组测试数据：

每组数据一行一个正整数 n $(1 \le n \le 10^{18})$，表示荷叶的数量。

输出

对于每组数据，输出一行一个正整数表示前往第 n 片荷叶的方案数。

解法

考虑输入数据的范围，可以先打表找到规律，再尝试证明。记 n 对应输出为 $f(n)$，可发现

$$f(n) = \begin{cases} 1, & n = 1 \\ \sum_{1 \le i \le n-1, i \oplus n < n} f(i), & n \ge 2 \end{cases}$$

可先用 Python 或 Rust 等求出前 32 项：

```
n = 32
ans = [0] * (n + 1)
ans[1] = 1

for i in range(1, n + 1):
    for j in range(1, i):
        if ans[j] and i ^ j < i:
            ans[i] += ans[j]

for i, j in enumerate(ans):
    print(f"{i}: {j}")
```

```rust
fn main() {
    let n = 32;
    let mut ans = vec![0; n + 1];
    ans[1] = 1;
    for i in 1..n + 1 {
        for j in 1..i {
            if ans[j] > 0 && i ^ j < i {
                ans[i] += ans[j];
            }
        }
    }
    ans.iter().enumerate().for_each(|(i, x)| println!("{i}: {x}"));
}
```

输出为：

```
0: 0
1: 1
2: 0
3: 1
4: 0
5: 1
6: 2
7: 5
8: 0
9: 1
10: 2
11: 5
12: 16
13: 33
14: 66
15: 133
16: 0
17: 1
18: 2
19: 5
20: 16
21: 33
22: 66
23: 133
24: 512
25: 1025
26: 2050
27: 4101
28: 8208
29: 16417
30: 32834
31: 65669
32: 0
```

进一步思考可发现，对正整数 n，小于 n 且与 n 异或也小于 n 的数，其二进制最高位在 n 中为 1。因此对 n 来说，答案是它的二进制表示为 1 的位，在 1~$n-1$ 中最高位是这一位的答案和（相当于一个 DP）。据此可写出 $O(n)$ 的做法。

发现该数列在 2 的整数次幂之间存在规律性，可得到更高效的做法：

1. 若 n 为 2 的正整数次幂，则 $f(n)=0$；$f(1)=1$。
2. 否则，将二进制表示的最高位的 1 去除后，调用递归函数 rec 可得答案。
   $rec(n)=2^{n+(n\text{二进制表示的最高位位数})-2}+rec(n\text{ 二进制表示的最高位的 1 去除后的数})$。

> 标程中的规律为：
>
> 记 s_n 为 n 的二进制表示，$s_n[i]$ 表示 n 的二进制表示的第 i 位，同时我们记 h_n 表示 n 的二进制表示的最高位（从 0 开始计）。则我们有 $ans[n]=\sum_{i=1}^{n-1}s_n[h_i]\cdot ans[i]$。从而我们可以给出如下两个结论：
>
> 若 $s_n[h_n-1]=0$，则 $ans[n]=ans[n-2^{h_n-1}]$；
>
> 若 $s_n[h_n-1]=1$，则 $ans[n]=2^{2^{h_n-1}+(h_n-3)+(n-2^{h_n}-2^{h_n-1})}+ans[n-2^{h_n}]$。
>
> （最后附上助教gg的证明思路）

代码

```
#include <iostream>
using namespace std;
const long long MOD = 998244353ll;
long long binpow(long long a, long long b, long long p)
{
    long long res = 1ll;
    for (a %= p; b; b >>= 1ll)
    {
        if (b & 1ll)
        {
            res = res * a % p;
        }
        a = a * a % p;
    }
    return res;
}
long long rec(long long n)
{
    if (n == 0ll || n == 1ll)
    {
        return n;
    }
    long long i = 63ll;
    while (!(n & (1ll << i)) && i >= 0)
    {
        --i;
    }
    return (binpow(2ll, n + i - 2ll, MOD) + rec(n & ((1ll << i) - 1ll))) % MOD;
}
long long solve(long long n)
{
    if (n == 1ll)
    {
        return 1ll;
    }
    long long i = 63ll;
```

```
    while (!(n & (1ll << i)) && i >= 0)
    {
        --i;
    }
    return ~i ? rec(n & ((1ll << i) - 1ll)) : 0ll;
}
int main()
{
    ios::sync_with_stdio(false);
    cin.tie(0);
    long long t, n;
    cin >> t;
    while (t--)
    {
        cin >> n;
        cout << solve(n) << endl;
    }
    return 0;
}
```

首先我们给出一些符号标记定义如下：

记 s_n 为 n 的二进制表示，$s_n[i]$ 表示 n 的二进制表示的第 i 位，同时我们记 h_n 表示 n 的二进制表示的最高位（从 0 开始计）。例如 $s_6[2]=s_6[1]=1, s_6[0]=0, h_6=2$。

则我们有

$$ans[n]=\sum_{i=1}^{n-1} s_n[h_i]\cdot ans[i]$$

从而我们可以给出如下两个结论：

- 若 $s_n[h_n-1]=0$，则 $ans[n]=ans[n-2^{h_n-1}]$。

这是因为对任意的 $h\ (h\geq 1)$，当 $n=2^h$ 时，显然有 $ans[n]=ans[\frac{n}{2}]=ans[n-2^{h-1}]=0$。则对任意 $2^h<n<2^h+2^{h-1}$，我们有

$$ans[n]=\sum_{i=1}^{n-1} s_n[h_i]\cdot ans[i]=\sum_{i=2^h}^{n-1} ans[i]+\sum_{i=1}^{2^{h-1}-1} s_n[h_i]\cdot ans[i]$$

又对任意的 $2^{h-1}<n<2^h$，我们有

$$ans[n]=\sum_{i=1}^{n-1} s_n[h_i]\cdot ans[i]=\sum_{i=2^{h-1}}^{n-1} ans[i]+\sum_{i=1}^{2^{h-1}-1} s_n[h_i]\cdot ans[i]$$

从而我们有 $ans[n]=ans[n-2^{h_n-1}]$。

- 若 $s_n[h_n-1]=1$，则 $ans[n]=2^{2^{h_n-1}+(h_n-3)+(n-2^{h_n}-2^{h_n-1})}+ans[n-2^{h_n}]$。

首先我们记 $sum[h]=\sum_{i=2^h}^{2^{h+1}-1} ans[i]$，则

$$\begin{aligned}
ans[n]&=\sum_{i=1}^{n-1} s_n[h_i]\cdot ans[i]\\
&=\sum_{i=1}^{2^{h_n-1}-1} s_n[h_i]\cdot ans[i]+\sum_{i=2^{h_n-1}}^{n-2^{h_n}-1} ans[i]+sum[h_n-1]\cdot 2^{(n-2^{h_n}-2^{h_n-1})+1}\\
&=ans[n-2^{h_n}]+sum[h_n-1]\cdot 2^{(n-2^{h_n}-2^{h_n-1})+1}
\end{aligned}$$

下面我们只需证明 $sum[h]=2^{2^h+(h-3)}\ (h>0)$。

这是因为 $sum[h+1]=sum[h]\cdot(1+\sum_{i=0}^{h}2^i)=sum[h]\cdot 2^{h+1}$，从而即有 $sum[h]=2^{2^h+(h-3)}$。

代入上式即得 $ans[n]=2^{2^{h_n-1}+(h_n-3)+(n-2^{h_n}-2^{h_n-1})}+ans[n-2^{h_n}]$。