Gaitonde and even bits

This question can be solved using bit manipulation. For any n we can run a loop from 1 to n and for every number A between 1 to n we can count number of set bits in it. If number of set bits are even then increase the main count by 1 and check for next number.

For calculating answer for 1 to n we can use the data of 1 to n-1 in following manner.

If number of se bits in n is odd the dp[n] = dp[n-1] , otherwise dp[n] = dp[n-1]+1.

So, in this way we can precalculate answer for every number between 1 to 100000.

Time complexity: - O(n)

Relevant links: -

https://www.geeksforgeeks.org/count-set-bits-in-an-integer/

https://www.geeksforgeeks.org/number-integers-odd-number-set-bits/

C++ solution

```
#include <iostream>

#include <fstream>

#include <fstream>

#define File freopen("input7.txt" , "r" , stdin); freopen("output7.txt" , "w" , stdout)

using namespace std;

// funtion for checking number of even bits

bool even(int n){

  int c = 0 ;

  while(n!=0){

    c+=n&1;

    n = n>>1;

  }

  return c%2==0;

}

int main () {

  File;
```

```
  // array for precalculating count for 1 to 100000
  int dp[100001];
  dp[0] = 0;
  for(int i = 1 ; i<100001 ; i++){
    dp[i] = dp[i-1];
    if(even(i)) dp[i]++;
  }
  int test , n;
  cin>>test;
  while(test--){
    cin>>n;
    cout<<dp[n]<<'\n';
  }
  return 0;
}
```