```
int add(int a, int b)
{
 int res = a + b;
 if(res >= MOD)
  return res - MOD;
 return res;
}

int mult(int a, int b)
{
 long long res = a;
 res *= b;
 if(res >= MOD)
  return res % MOD;
 return res;
}

struct matrix
{
 int arr[SZ][SZ];

 void reset()
 {
  memset(arr, 0, sizeof(arr));
 }

 void makeiden()
 {
  reset();
  for(int i=0;i<SZ;i++)
  {
   arr[i][i] = 1;
  }
 }

 matrix operator + (const matrix &o) const
 {
  matrix res;
  for(int i=0;i<SZ;i++)
  {
   for(int j=0;j<SZ;j++)
   {
    res.arr[i][j] = add(arr[i][j], o.arr[i][j]);
   }
  }
  return res;
 }

 matrix operator * (const matrix &o) const
 {
  matrix res;
  for(int i=0;i<SZ;i++)
  {
   for(int j=0;j<SZ;j++)
```

```
   {
    res.arr[i][j] = 0;
    for(int k=0;k<SZ;k++)
    {
     res.arr[i][j] = add(res.arr[i][j] , mult(arr[i][k] , o.arr[k][j]));
    }
   }
  }
  return res;
 }
};

matrix power(matrix a, int b)
{
 matrix res;
 res.makeiden();
 while(b)
 {
  if(b & 1)
  {
   res = res * a;
  }
  a = a * a;
  b >>= 1;
 }
 return res;
}
```