Number Theory Lecture 9

Tuesday, 16 July 2024 6:05 AM

$$\sum_{k=0}^{2} A[i][k] * B[k][j]$$

Fast Matrix Exponentiation

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

$$A^2 = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} a^2+bc & ab+bd \\ ac+cd & bc+d^2 \end{bmatrix}$$

ij: i → row of 1st, j → column of 2nd

$$A^n = \underbrace{A * A * A * A \cdots * A}_{n \text{ times}} \qquad O(n)$$

$$A^n = \begin{cases} A^{n/2} * A^{n/2}, & \text{if } n \text{ is even} \\ A^{n/2} * A^{n/2} * A, & \text{if } n \text{ is odd} \\ I, & \text{if } n == 0 \\ A, & \text{if } n == 1 \end{cases} \qquad O(\log n)$$

← base case.

$I * A = A$ ✓

```
typedef struct {
    ll m[2][2];
} matrix;
matrix identity() {
    matrix I = {{
        {1, 0},
        {0, 1}
    }};
    return I;
}
matrix mul(matrix &a, matrix &b) {
    matrix c = {{
        {0, 0},
        {0, 0}
    }};
    FOR(i,0,2) {
        FOR(j,0,2) {
            FOR(k,0,2) {
                c.m[i][j] = (c.m[i][j]+a.m[i][k]*b.m[k][j])%MOD;
            }
        }
    }
    return c;
}
matrix pow(matrix &a, int b) {
    if(b==0) return identity();
    matrix ans = pow(a, b/2);
    ans = mul(ans, ans);
    if(b%2!=0) ans = mul(ans, a);
    return ans;
}
void solve() {
    matrix a = {{
        {2, 3},
        {1, 2}
    }};
    matrix b = pow(a, 6);
    cout << b.m[0][0] << " " << b.m[0][1] << endl;
    cout << b.m[1][0] << " " << b.m[1][1] << endl;
}
```

Why Matrix Exponentiation ?

$$\begin{bmatrix} 2 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 4 \\ 3 \end{bmatrix} = \begin{bmatrix} 5 \\ 7 \end{bmatrix}$$

Transform matrix

$$\begin{bmatrix} 2 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 5 \\ 7 \end{bmatrix} = \begin{bmatrix} 3 \\ 12 \end{bmatrix}$$

$$\begin{bmatrix} 2 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 3 \\ 12 \end{bmatrix} = \begin{bmatrix} -6 \\ 15 \end{bmatrix}$$

$4\hat{i} + 3\hat{j}$

$(2x-y,\ x+y)$

$x \quad y$

$(4, 3)$

$(5, 7)$

$\boxed{(3, 12)}$

$(-6, 15)$

$\vdots$

$$\begin{bmatrix} 2 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 2x-y \\ x+y \end{bmatrix}$$

Initial

$(4, 3)$

$(x, y) \xrightarrow{100 \text{ times}} (2x-y,\ x+y)$

$$\underbrace{\begin{bmatrix} 2 & -1 \\ 1 & 1 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} 4 \\ 3 \end{bmatrix}}_{B}$$

$\ldots A(A(AB))$

$= A^n B$

$$A^2 = \begin{bmatrix} 2 & -1 \\ 1 & 1 \end{bmatrix}\begin{bmatrix} 2 & -1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 3 & -3 \\ 3 & 0 \end{bmatrix}$$

$$A^2 \begin{bmatrix} 4 \\ 3 \end{bmatrix} = \begin{bmatrix} 3 & -3 \\ 3 & 0 \end{bmatrix}\begin{bmatrix} 4 \\ 3 \end{bmatrix} = \begin{bmatrix} 3 \\ 12 \end{bmatrix}$$

$A^n \rightarrow \underline{\underline{\log}}\, n$

=x=

① n^{th} fibonacci number

$$\boxed{\begin{matrix} F_0 = \underline{\underline{1}} \\ F_1 = \underline{\underline{1}} \end{matrix}}$$

$$\underline{\underline{F_k}} = \underline{\underline{F_{k-1}}} + \underline{\underline{F_{k-2}}}$$

y x

1, 1, 2, 3, 5, 8, 13, ...

y x

y x

y x

...

x , y

(1, 1)

(2, 1)

(3, 2)

(5, 3)

(x, y)

$\underline{\underline{(x+y,\ x)}}$

$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x+y \\ x \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} F_{k-1} \\ F_{k-2} \end{bmatrix} = \begin{bmatrix} F_k \\ F_{k-1} \end{bmatrix}$$

$F_0 = 1, \quad F_1 = 1 \quad F_2 = 2$

$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} F_1 \\ F_0 \end{bmatrix} = \begin{bmatrix} F_2 \\ F_1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^1 \to 2 \qquad \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$F_3. \quad \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^2 = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix} \to 3$$

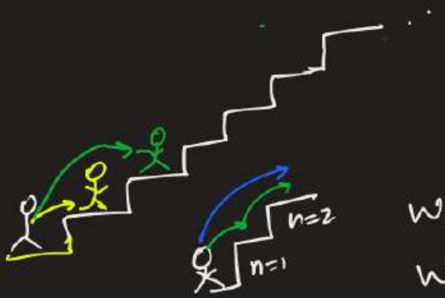
$$F_4. \quad \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^3 = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 3 & 2 \\ 2 & 1 \end{bmatrix} \to 5$$

$$\vdots$$

```
typedef struct {
    ll m[2][2];
} matrix;
matrix identity() {
    matrix I = {{
        {1, 0},
        {0, 1}
    }};
    return I;
}
matrix mul(matrix &a, matrix &b) {
    matrix c = {{
        {0, 0},
        {0, 0}
    }};
    FOR(i,0,2) {
        FOR(j,0,2) {
            FOR(k,0,2) {
                c.m[i][j] = (c.m[i][j]+a.m[i][k]*b.m[k][j])%MOD;
            }
        }
    }
    return c;
}
```

```cpp
}
matrix pow(matrix &a, int b) {
    if(b==0) return identity();
    matrix ans = pow(a, b/2);
    ans = mul(ans, ans);
    if(b%2!=0) ans = mul(ans, a);
    return ans;
}
void solve() {
    matrix a = {{
        {1, 1},
        {1, 0}
    }};
    int n;
    cin >> n;
    if(n == 0)  cout << 1 << endl;
    if(n == 1)  cout << 1 << endl;
    matrix pa = pow(a, n-1);
    cout << pa.m[0][0] + pa.m[0][1] << endl;
}
```

https://leetcode.com/problems/climbing-stairs/description/



1, 2

$$\boxed{W_n = W_{n-1} + W_{n-2}}$$

$W_1 = 1$

$W_2 = 2$

1 1
2

$\{$ 1 1 1
1 2
2 1

$W_3 = W_2 + W_1$

$W_4 = W_3 + W_2$

$\vdots$

$W_1 = 1$
$W_2 = 2$ ✓

$W_0 = 1$
$W_1 = 1$ ✓

$$W_n = W_{n-1} + W_{n-2}$$

$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \underline{W_{n-1}} \\ \underline{W_{n-2}} \end{bmatrix} = \begin{bmatrix} \underline{W_n} \\ \underline{W_{n-1}} \end{bmatrix}$$

W_{n-4} $\qquad\qquad$ W_{n-3} $\qquad\qquad$ W_{n-2}

$W_{n-5} \longrightarrow$ $\quad$ $W_{n-4} \longrightarrow$ $\quad$ $W_{n-3} \longrightarrow \cdots$

$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} W_2 = 2 \\ W_1 = 1 \end{bmatrix}$$

(3) → 1 time

(4) → 2 times

(n) → n−2 times

```python
def mul(a, b):
    c = []
    for i in range(len(a)):
        row = []
        for j in range(len(b)):
            row.append(0)
            for k in range(len(a)):
                row[-1] += a[i][k]*b[k][j]
        c.append(row)
    return c
def pow(a, b):
    if b==0:
        return [[1, 0], [0, 1]]
    ans = pow(a, b//2)
    ans = mul(ans, ans)
    if b%2!=0:
        ans = mul(ans, a)
    return ans
class Solution:
    def climbStairs(self, n: int) -> int:
        m = [[1,1],[1,0]]
        if n==1:
            return 1
        if n==2:
            return 2
        m = pow(m, n-2)
        return 2*m[0][0]+m[0][1]
```

```cpp
typedef struct {
    long long v11, v12, v21, v22;
} matrix;
matrix identitiy() {
    return matrix {1, 0, 1, 1};
}
matrix mul(matrix a, matrix b) {
    return matrix {a.v11*b.v11+a.v12*b.v21,
                   a.v11*b.v21+a.v12*b.v22,
                   a.v21*b.v11+a.v22*b.v21,
                   a.v21*b.v21+a.v22*b.v22};
}
matrix pow(matrix a, int b) {
    if(b==1)    return a;
    matrix ans = pow(a, b/2);
    ans = mul(ans, ans);
    if(b%2!=0)  ans = mul(ans, a);
    return ans;
}
class Solution {
public:
    int climbStairs(int n) {
        matrix fib = matrix{1, 1, 1, 0};
        if(n==1) return 1;
        if(n==2) return 2;
        matrix fibn = pow(fib, n-2);
        return 2*fibn.v11 + fibn.v12;
    }
};
```