**LAB [6], [2/15/2019] MCS 253P**

**Name: Yu Qin**

**General Problem Description**

Dijkstra’s SSSP:

N vertices numbered from 1 to n. For each vertex i, it has two undirected edge (i, i + 1) and (i, i\*3). Find the shortest path from 1 to n.

**Additional Problem Specifics**

Range of N: 1 <= N <= 1000

Range of test case number: 1 <= T <= 30

Special test case when N = 1, should output 0

**Proposed Algorithm**

F[i]: shortest path from 1 to i

F[1] = 0;

F[i] = f[i - 1] + 1 if i % 3 != 0

F[i] = min(f[i - 1], f[i / 3]) + 1 if i % 3 == 0

f[i / 3] <= f[i - 1] if i % 3 ==0

so f[i] = f[i / 3] if i % 3 == 0

Time complexity: O (T\*log3(n))

Space complexity: O(1)

#include <iostream>

#include <cstdio>

using namespace std;

int main() {

int T;

cin >> T;

if (T < 1 || T > 30) {

printf("Number of test case should be in [1, 30].\n");

return -1;

}

for (int i = 0; i < T; i ++) {

int n;

cin >> n;

if (n < 1 || n > 1000) {

printf("Number of vertices should be in [1, 1000].\n");

return -1;

}

int ans = 0;

while (n > 1) {

if (n % 3 == 0) {

ans ++;

n /= 3;

}else {

ans += n % 3;

n -= n % 3;

}

}

ans -= (n == 0);

cout << ans << endl;

}

return 0;

}