/\*

Standard implementation of the Floyd-Warshall Algorithm

\*/

#include <stdio.h>

#include <stdlib.h>

#include <sys/time.h>

void graph\_init\_random(int \*\*adjm, int seed, int n, int m)

{

unsigned int i, j;

srand(seed);

for(i=0; i<n; i++)

for(j=0; j<n; j++)

adjm[i][j] = abs(((int)rand()) % 1048576);

for(i=0; i<n; i++)adjm[i][i]=0;

}

int min(int a, int b)

{

if(a<=b)return a;

else return b;

}

int main(int argc, char \*\*argv)

{

int \*\*A;

int i,j,k;

struct timeval t1, t2;

double time;

int N=1024;

if (argc != 2) {

fprintf(stdout,"Usage: %s N\n", argv[0]);

exit(0);

}

N=atoi(argv[1]);

A = (int \*\*) malloc(N\*sizeof(int \*));

for(i=0; i<N; i++) A[i] = (int \*) malloc(N\*sizeof(int));

graph\_init\_random(A,-1,N,128\*N);

gettimeofday(&t1,0);

for(k=0;k<N;k++)

#pragma omp parallel for shared(A) private(i,j) firstprivate(k, N)

for(i=0; i<N; i++)

for(j=0; j<N; j++)

A[i][j]=min(A[i][j], A[i][k] + A[k][j]);

gettimeofday(&t2,0);

time=(double)((t2.tv\_sec-t1.tv\_sec)\*1000000+t2.tv\_usec-t1.tv\_usec)/1000000;

printf("FW,%d,%.4f\n", N, time);

/\*

for(i=0; i<N; i++)

for(j=0; j<N; j++) fprintf(stdout,"%d\n", A[i][j]);

\*/

return 0;

}