```
#include "rk4_solver.hh"
#include <cstdlib>
#include <ctime>
#include <cstdio>
#include <iostream>
#include <math.h>
#include <omp.h>

int main(){
  double T_final = 100.0;
  double hi_step = 1.0;
  double tolerance = 0.000001;
  double J = 0.1;
  double K = 1;
  double N_intrvls = 2;
  rk4 myRk4_1(1250, hi_step, tolerance, J, K, N_intrvls,0.7,1);
  double t1 = omp_get_wtime();
  myRk4_1.compute_solution(T_final);
  double t2 = omp_get_wtime();
  myRk4_1.terminate();


  // Produce results for the comparison between
  // Initial code and Barnes-Hut enabled code
  // for an increasing number of swarmalators N
  for (double N_pwr=4.10; N_pwr<4.15; N_pwr += 0.05 ){
    int NN = (int) pow(10, N_pwr);
    rk4 myRk4_norm(NN, hi_step, tolerance, J, K, N_intrvls,0.1,0);
    double t1 = omp_get_wtime();
    myRk4_norm.compute_solution(T_final);
    double t2 = omp_get_wtime();
    myRk4_norm.terminate();
    rk4 myRk4_bh(NN, hi_step, tolerance, J, K, N_intrvls,0.1,1);
    double t3 = omp_get_wtime();
    myRk4_bh.compute_solution(T_final);
    double t4 = omp_get_wtime();
    myRk4_bh.terminate();
    printf("%d %f %f\n",NN, t2-t1, t4-t3);
  }

  // Produce statistically significant results for
  // Barnes Hut with adaptive timestep as a function
  // of theta_BH.
  for (double pwr=-2; pwr<-0.2; pwr+=0.02){
    double theta = pow(10,pwr);
    rk4 myRk4_1(2000, hi_step, tolerance, J, K, N_intrvls,theta,1);
    double t1 = omp_get_wtime();
    myRk4_1.compute_solution(T_final);
    double t2 = omp_get_wtime();
    myRk4_1.terminate();
    rk4 myRk4_2(2000, hi_step, tolerance, J, K, N_intrvls,theta,1);
    double t3 = omp_get_wtime();
    myRk4_2.compute_solution(T_final);
    double t4 = omp_get_wtime();
    myRk4_2.terminate();
    rk4 myRk4_3(2000, hi_step, tolerance, J, K, N_intrvls,theta,1);
    double t5 = omp_get_wtime();
    myRk4_3.compute_solution(T_final);
    double t6 = omp_get_wtime();
    myRk4_3.terminate();
    rk4 myRk4_4(2000, hi_step, tolerance, J, K, N_intrvls,theta,1);
    double t7 = omp_get_wtime();
    myRk4_4.compute_solution(T_final);
    double t8 = omp_get_wtime();
    myRk4_4.terminate();
    rk4 myRk4_5(2000, hi_step, tolerance, J, K, N_intrvls,theta,1);
    double t9 = omp_get_wtime();
    myRk4_5.compute_solution(T_final);
    double t10 = omp_get_wtime();
    myRk4_5.terminate();
    printf("%f %f %f %f %f %f\n",theta,t2-t1,t4-t3,t6-t5,t8-t7,t10-t9);
  }
```

```
  printf("Total Computation Time: %f\n", t2-t1);
  return 0;
}
```