```cpp
//OpenMP.cpp : 此文件包含 "main" 函数。程序执行将在此处开始并结束。
//
/*
#include <iostream>

int main()
{
#pragma omp parallel
    {
        std::cout << "Hello World!\n";
    }
}*/

#include <iostream>
#include<iomanip>
#include <stdio.h>
#include <omp.h>
using namespace std;
const int thread_num = 4;                //线程数
const int n = 200000;                     //划分区间数:越大值越精确
int main()
{
    int tid;
    double step = 1.0 / (double)n;
    double tid_sum[thread_num]{};
    double pi = 0.0;
    omp_set_num_threads(thread_num);
#pragma omp parallel private(tid)        //设置tid为各个线程私有
    {                                    //并行域开始,每个线程都会执行该代码
        double x;
        tid = omp_get_thread_num();      //tid用omp_get_thread_num()赋值为线程编号
        tid_sum[tid] = 0.0;
        for (int i=tid;i<n;i+=thread_num)
        {
            x = (i + 0.5) * step;
            tid_sum[tid] += 4.0 / (x * x + 1.0);
        }

        cout << "tid=" << tid << "," <<
            "tid_sum[" << tid << "]=" <<
            setprecision(20)<<tid_sum[tid] * step<< endl;
    }
    for(int i=0;i<thread_num;i++)
        pi += tid_sum[i];
    pi *= step;
    cout<<setprecision(20)<<"pi="<<pi<<endl;
}

/*
#include <iostream>
#include <time.h>
#include <math.h>
#include "omp.h"
using namespace std;

void test_fn(int epochs, int num_thread) {
```

```
57      omp_set_num_threads(num_thread);
58      double c = 0;
59      clock_t start = clock();
60  #pragma omp parallel for
61      for (int i = 0; i < epochs; i++) {
62          //随便做一个无用的计算
63          c += exp(1) * exp(1) * log(2);
64      }
65      clock_t end = clock();
66      cout << num_thread << "线程：用时" << end - start << ends << "循环次数：" <<
          epochs << endl;
67  }
68
69  int main()
70  {
71      test_fn(10000000, 1);
72      test_fn(10000000, 2);
73      test_fn(10000000, 4);
74      test_fn(10000000, 8);
75      test_fn(10000000, 12);
76      return 0;
77  }*/
78
79  // 运行程序: Ctrl + F5 或调试 >“开始执行(不调试)”菜单
80  // 调试程序: F5 或调试 >“开始调试”菜单
81
82  // 入门使用技巧:
83  //   1. 使用解决方案资源管理器窗口添加/管理文件
84  //   2. 使用团队资源管理器窗口连接到源代码管理
85  //   3. 使用输出窗口查看生成输出和其他消息
86  //   4. 使用错误列表窗口查看错误
87  //   5. 转到“项目”>“添加新项”以创建新的代码文件，或转到“项目”>“添加现有项”以将
      现有代码文件添加到项目
88  //   6. 将来，若要再次打开此项目，请转到“文件”>“打开”>“项目”并选择 .sln 文件
89
```