专业考核加分题：马葱的依赖赋值

马葱在开发某程序时，被甲方要求对一个算法进行并行化。该算法遍历数组里的每一个元素，并该元素的值更新为该数组里以该元素为下标的元素的值。即：a[i]=a[a[i]]。假设遍历到第 1000 个元素，该元素的原值是 123，那么 a[1000]=a[a[1000]]=a[123]，即将 a[123]的值赋给 a[1000]。

并行化的目的，是为了在多核系统上加速本算法的同时，还能做到与串行计算结果的一致。问题就出现了：因为取值是有先后顺序的，并行计算的同时，一个核心如何确保它的取值是符合顺序的呢？

请使用 OpenMP 在多核系统上加速本算法。只能修改两个/**********/注释符号内包围的代码。

一、开发环境的配置

请下载并安装的 Embarcadero_Dev-Cpp。可以在群文件里找到，也可使用以下下载地址：

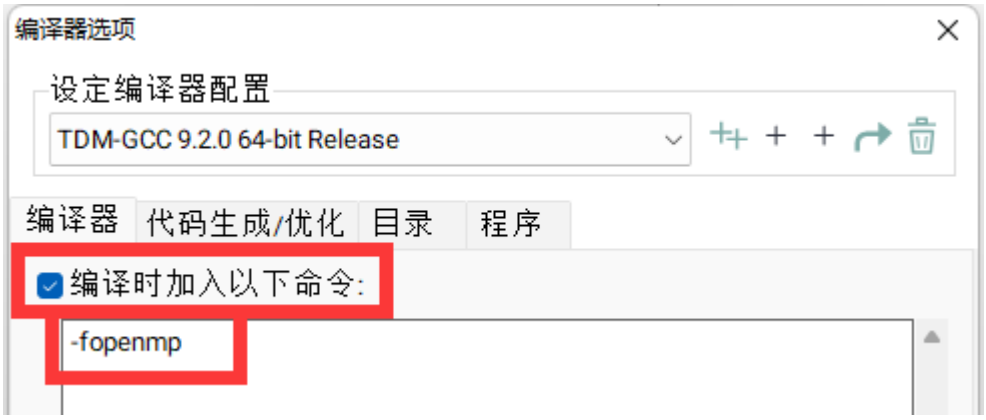
阿里云 https://www.aliyundrive.com/s/Rx6MDYqc3N8

百度云 https://pan.baidu.com/s/1eXebXcf-UQApItOeEbeYGA?pwd=3uft

官网下载（无梯巨慢）https://github.com/Embarcadero/Dev-Cpp/releases/download/v6.3/Embarcadero_Dev-Cpp_6.3_TDM-GCC_9.2_Setup.exe

该编辑器集成了较新的 TDM-GCC 版本。安装完成后，打开工具-编译选项：

勾选“编译时加入以下命令”，并在下方加入“-fopenmp”。



现在，你的 DEV-C++就支持 OpenMP 啦，是不是很简单？如果你使用其它编辑器，请百度搜索“编辑器名称+OpenMP”就可以找到配置教程了。

二、参考资料

- 建议参考《并行程序设计导论》（附件有电子版）第一、二、五章。其它章节可以跳过，没有相关性。如果时间实在是紧，可以只看第五章。
- 压缩包里，还附带了一段视频，为对算法的解读。
- 如果时间充裕，可以看看新竹清华大学的“并行计算与并行编程课程”。地址：https://www.bilibili.com/video/BV1Yt411W7td

逐行代码解读

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <omp.h>

#define N 100000000
```

这个宏定义，决定了运算规模的大小。当你在调试的时候，可以把它改小一些，以减少调试的时间。

```
int main()
{
    int *a, *b;
    a = (int*)malloc(sizeof(int) * N);
    b = (int*)malloc(sizeof(int) * N);
    if(!a || !b){
        printf("malloc failed!\n");
        exit(0);
    }
```

分配 a 和 b 两个数组，并分配空间。其中，a 用来存储正确答案，b 用来存储并行计算的答案

```
    srand((unsigned int)time(NULL));
    for(int i = 0; i < N; i++) a[i] = b[i] = rand() % N;
```

为 a 和 b 两个数组随机赋初始值，两个数组的数值相等

```
    double start1, end1, start2, end2;
```

用来记录时间的变量

```
    start1 = omp_get_wtime();
```

a 数组运算计时开始

```
    for(int i = 0; i < N; i++){
        a[i] = a[a[i]];
    }
```

对 a 数组执行操作

```
    end1 = omp_get_wtime();
```

a 数组运算计时结束

```
    start2 = omp_get_wtime();
```

b 数组运算计时开始

```
/******************/
    // write your code here
    for(int i = 0; i < N; i++){
        b[i] = b[b[i]];
```

```
    }
/******************/
```
只有上述两个注释符号/******************/间的代码可以修改哦

```
    end2 = omp_get_wtime();
```
b 数组运算计时结束

```
    int flag = 1;
```
用来记录结果是否正确的变量

```
    for(int i = 0; i < N; i++){
        if(a[i] != b[i]) {
            flag = 0;
            printf("Error on index %d, a[%d] = %d b[%d] = %d\n", i, i, a[i], i, b[i]);
            break;
        }
    }
```
a、b 比对结果是否一致，如果不一致，给出不一致的地方

```
    double t1, t2;
    t1 = end1 - start1;
    t2 = end2 - start2;
    if(flag) printf("The anwser is right!\n t1 = %lf\n t2 = %lf\n speedup = %lf\n", t1, t2, t1 / t2);
```
如果答案正确，那么给出你的加速比（恭喜你，你把程序加速了这么多倍!）

```
    return 0;
}
```