Lab 2：Stencil

实验目标

使用`stencil`模式求解热传导方程。

给定一个函数`u(t, x, y)`和时刻`T`。

你的目标是求解出`T`时刻的热分布，即`u(T, x, y)`。

框架代码

在二维平面的热传导方程如下：

$$\frac{\partial u(t,x,y)}{\partial t} = \alpha(\frac{\partial^2 u(t,x,y)}{\partial x^2} + \frac{\partial^2 u(t,x,y)}{\partial y^2})$$

对以上公式进行离散化，得到

$$u(t+1,x,y) - u(t,x,y) = \alpha[u(t,x+1,y) - 2u(t,x,y) + u(t,x-1,y) + u(t,x,y+1) - 2u(t,x,y) + u(t,x,y-1)]$$

其中`α`为导热系数。

根据以上公式你需要实现以下接口：

```
void student_stencil(double *u, int alpha, int T, int N, double *u_ans);
```

其中`u`是开始时刻即`t = 0`时的热分布，`alpha`是导热系数，`T`是最终时刻，`N`代表所需求解的区域大小， 你需要将最终答案存入`u_ans`中。

1. 为了方便起见，我们假定整个平面大小为`(N + 2) * (N + 2)`，但你只需要求解`N * N`大小平面上的$u(t,x,y)$，其中$x \in [1,N], y \in [1,N]$。在此区域外的热量均视为0，即：

   $$\forall x, \forall t, u(t,x,0) = u(t,x,N+1) = 0 \text{且} \forall y, \forall t, u(t,0,y) = u(t,N+1,y) = 0$$

   根据上述假定，我们为`u`和`u_ans`各申请了`(N + 2) * (N + 2)`大的空间。

   因此，你可以通过表达式`u[x * (N + 2) + y]`来访问平面上的`(x, y)`位置，

   同样， 你需要将最终`(x, y)`处的结果存入`u_ans[x * (N + 2) + y]`。

2. 此外，我们假定初始时刻为`0`，即给定的`u`表示`u(0, x, y)`，最终`u_ans`中保存的结果应该能够表示`u(T, x, y)`。
3. 实验文件可在`elearning`上下载, 其中包括`stencil.c`和`makefile`两个文件

   在`stencil.c`中，我们给出了一个框架，其中包括测试代码以及简单串行实现，你要做的是将`student_stencil`中的`naive_stencil`替换成自己实现的并行`stencil`代码。

运行方式

1.登陆服务器

我们为大家准备了一个24核的服务器，进入校内网后（校外需要使用VPN），使用命令行或 SSH 相关工具登入服务器。命令行 SSH 指令：

```
ssh user[your-student-ID]@10.20.26.203 # e.g. ssh user20307130000@10.20.26.203
```

初始密码为学号。

由于我们的服务器资源有限且程序对于性能十分敏感，因此禁止大家使用`vscode`或`clion`等软件远程连接服务器！！！如果发现，我们将会杀掉你的程序。

因此你应该在本地写好代码，并确保能够编译运行且结果正确后再将代码上传至服务器运行。

2.本地运行

编译:

本地编译代码需要安装 `gcc` 和 `make` 工具，使用 `windows` 系统的同学可以参考 GCC编译器的安装教程（Windows环境）_gcc安装教程-CSDN博客安装 `gcc`

安装 `make` 之后可以直接使用以下命令进行编译:

```
make
```

`windows` 下安装 `make` 可能较为麻烦, 你也可以直接使用以下命令进行编译:

```
gcc stencil.c -O3 -fopenmp -lm -Wall -o stencil
```

运行:

```
./stencil [T] [N] # T表示需要求解的时刻T, N表示空间大小
```

在本地运行代码会输出程序的正确性以及运行时间

PS: 如果你是 `Linux` 或 `Mac` 用户, 我相信你可以自行安装以上软件并完成编译.

3.上传至服务器

确保在本地能编译通过并且程序正确之后, 你可以将代码上传至服务器运行.

可以使用 `scp` 命令将整个 `lab2` 文件夹上传至服务器, 使用教程可以参考Linux scp命令 | 菜鸟教程 (runoob.com)。

登陆服务器后, 你可以先使用 `top` 命令查看是否有其他同学正在运行程序, 如果有, 建议你等待他的程序运行完成后再进行编译并运行. 由于我们的输入规模并不大, 一般最多也就等个十来分钟. 如果你发现某个同学的程序运行太长时间了, 请联系他或者助教将它停下

在服务器上编译及运行代码和本地一样

PS: 你可以不使用以上方式进行编译, 如果你发现任何编译器或编译选项能提高程序性能, 欢迎使用, 但请在实验报告中说明, 让助教知道怎么运行你的代码

评分标准

本次实验成绩组成：性能（80%） + 实验报告（20%）

首先，你必须答案正确，才能得分。

性能部分评分方式: 性能第一名80分, 第二名79分...以此类推.

实验文档中需要包含:

1.在服务器上的运行结果截图

2.算法主要思路

3.列举你使用的所有优化, 并简要说明它起作用的原因

4.(可选)你对于本次实验的想法, 任何方面都可以

实验报告不宜过长, 请勿在其中粘贴大量代码和注释.

另外，请勿修改框架代码，同学们唯一的任务是完成 `student_stencil` 函数。如果为了提升性能必须修改, 请在实验报告中说明. 如果框架代码存在问题，请及时反馈，助教将会尽快修正。

!注意: 严禁抄袭, 如果发现抄袭现象, 抄袭与被抄袭者本次实验均作0分处理

作业提交

作业截止时间: `2023年12月17日23点59分`

请在 `elearning` 上对应入口提交实验报告, 实验代码上传至服务器中你的用户目录下即可

请及时提交作业，如有迟交本次实验最高只有60分。