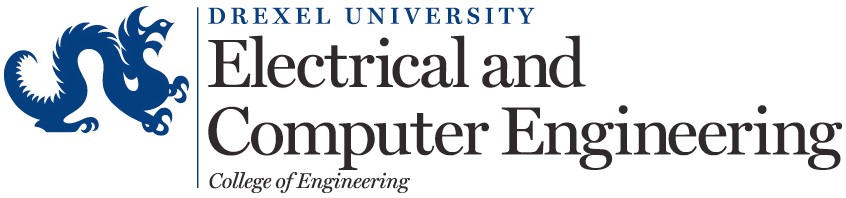
****

Drexel University

Electrical and Computer Engineering Dept.

ECEC-413

**Numerical Integration**

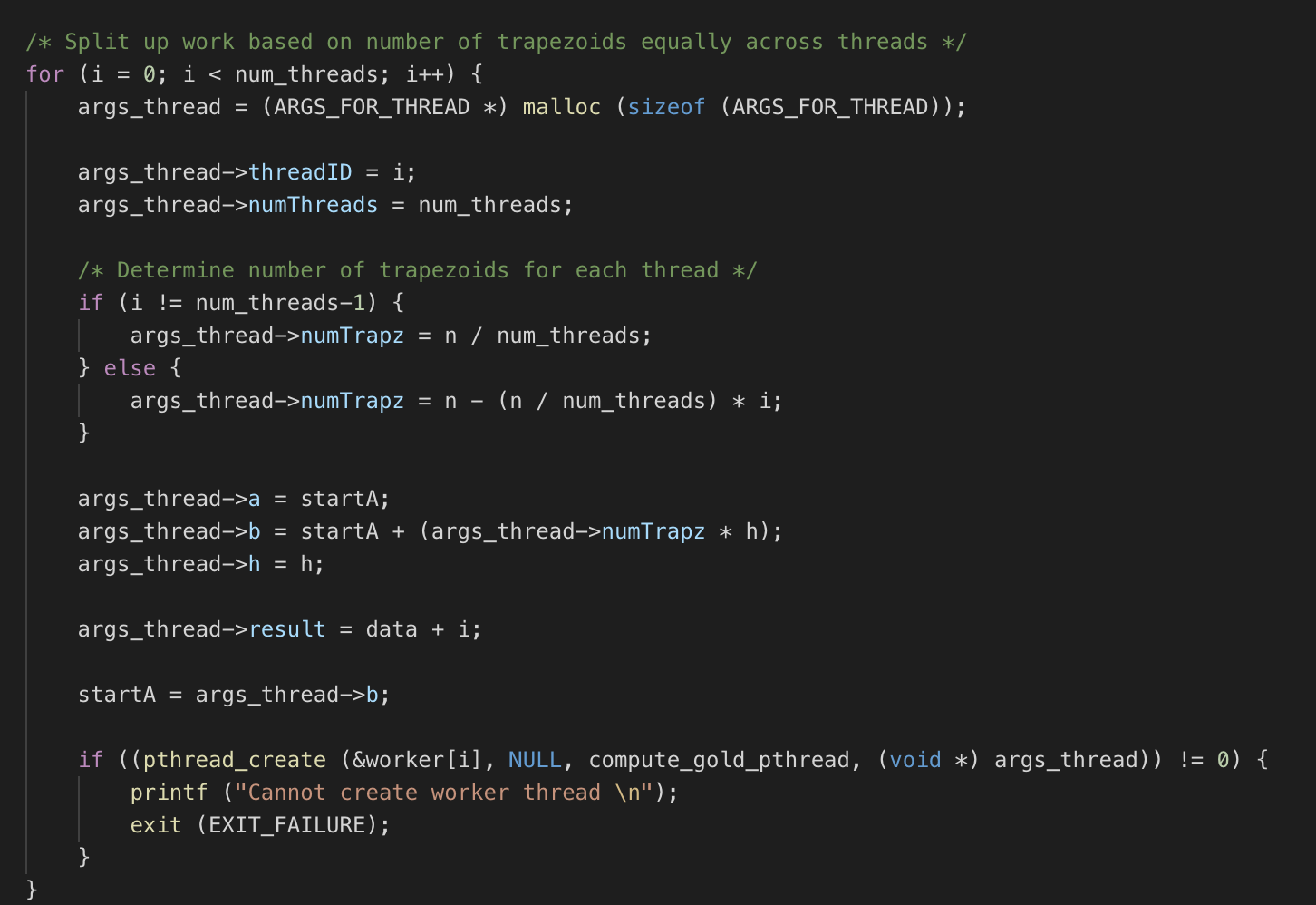
**Chris Kasper**

**Prof. Naga Kandasamy**

**DATE: 04/21/19**

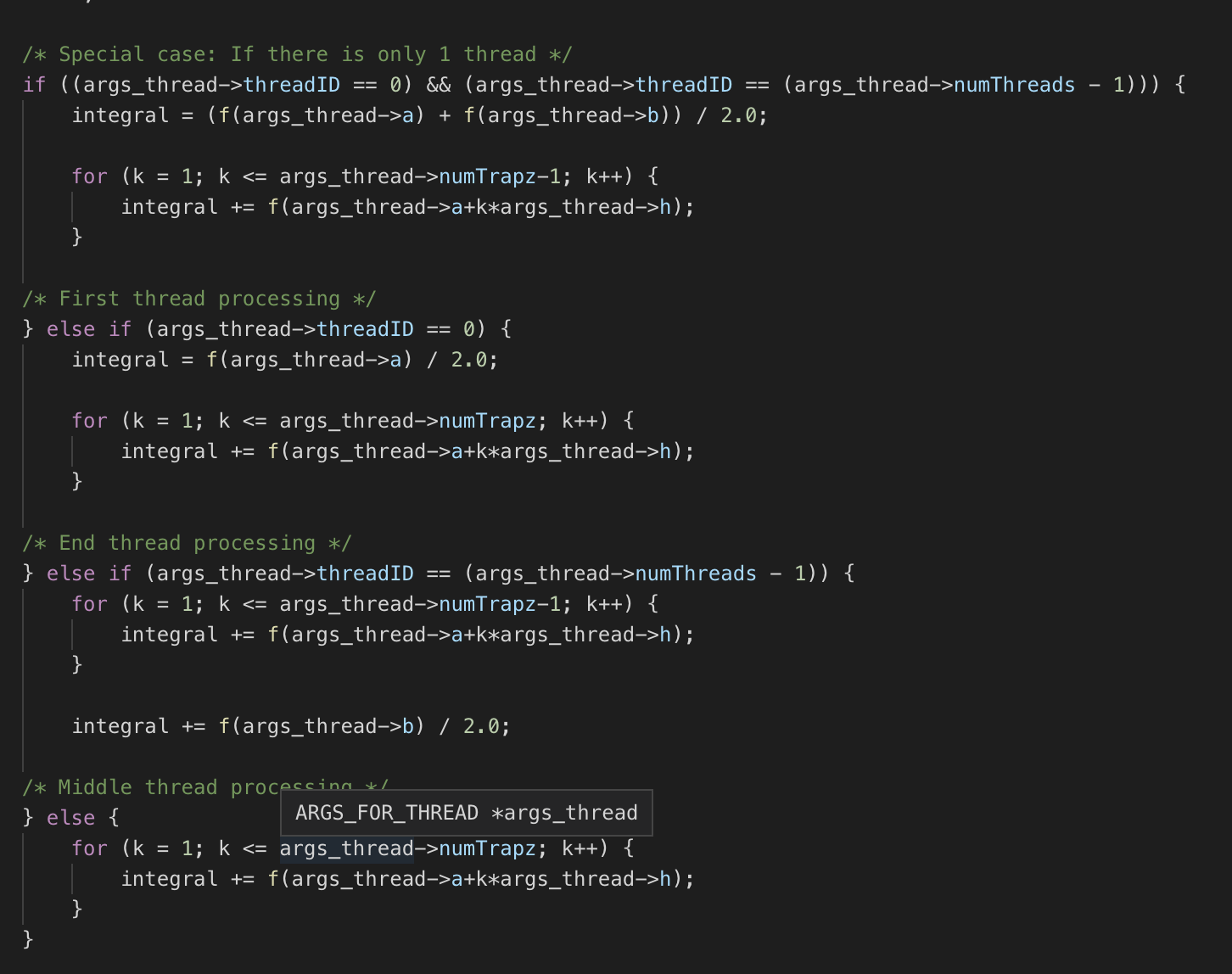
**Multi-threaded Design**

To extend the design of the numerical integration (using the trapezoid rule) to a multi-threaded design, the work has to be divided among the number of threads. This was done as such:



Essentially, each thread gets allocated a certain number of trapezoids being used to calculate the integral of the equation. Each thread has a start point, end point, and gets the number of trapezoids (including base of each trapezoid) it has to compute. The end result is stored in a data array structure, which each thread has a pointer to, so they can store their result. Each thread has a different pointer, so thread 0 goes into data[0], thread 1 goes into data[1], etc.

The thread function has certain checks to ensure the computation is done properly. There is different logic for the first thread, middle threads (not the first or last), and the end thread. There is even a special case thread in the case only 1 thread is being used. The logic is structured as such:



The thread function has certain checks to ensure the computation is done properly. There is different logic for the first thread, middle threads (not the first or last), and the end thread. There is even a special case thread in the case only 1 thread is being used.

Once all the threads finish, there is a loop over the data structure to calculate the final result.

**Speedup**

**Table 1: Speedup of integration on xunil-05**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Number of Trapezoids** | **Lower Limit** | **Upper Limit** | Single Thread | 2 Threads | 4 Threads | 8 Threads | 16 Threads |
| 100000 | 0 | 10 | 0.0052s | 0.0038s | 0.0024s | 0.0018s | 0.0020s |
| 100000 | 0 | 100 | 0.0052s | 0.0033s | 0.0022s | 0.0018s | 0.0027s |
| 1000000 | 0 | 10 | 0.0456s | 0.0253s | 0.0135s | 0.0075s | 0.0049s |
| 1000000 | 0 | 100 | 0.0465s | 0.0251s | 0.0140s | 0.0079s | 0.0046s |