Intro to Multi-Threading

Sam Shue, PhD

Demo Code and Slides:
https://github.com/SamShue/Intro-to-MultiThreading

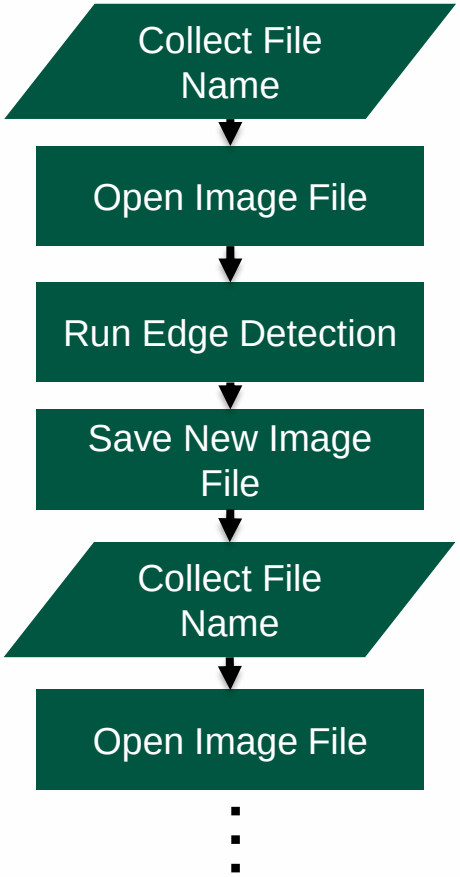
THE WILLIAM STATES LEE
COLLEGE OF ENGINEERING

Demo Code and Slides:
https://github.com/SamShue/Intro-to-MultiThreading

I have an application in C++ that extracts edges from several user-specified images.

Gaussian Blur
+
Laplacian
Edge
Detection

Current Application Flowchart

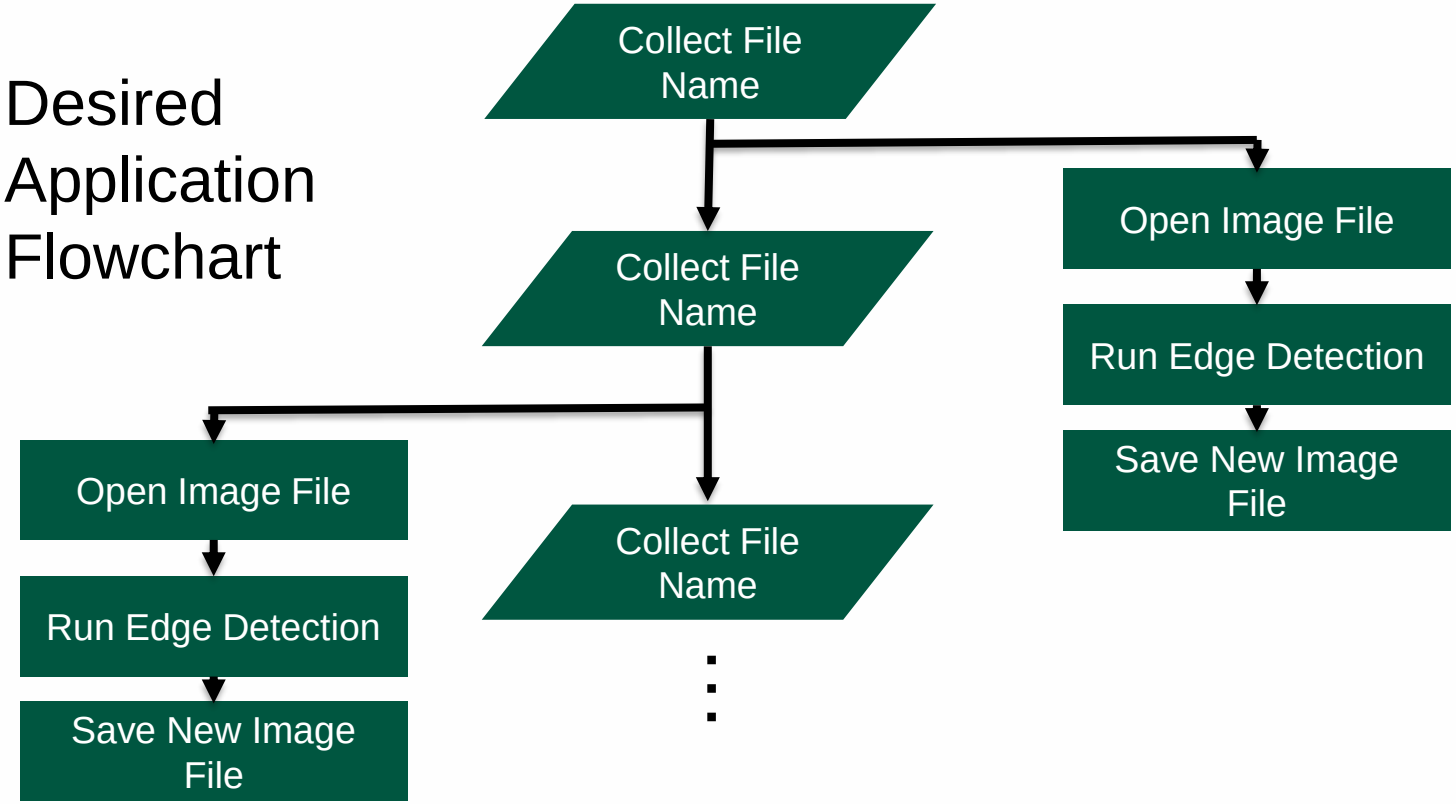


Application Demo

Problem: The process of detecting the edges takes a long time, and so does the user typing in the file name.

Desire: I would like to collect user input while the program also processes the image in the background

Desired Application Flowchart

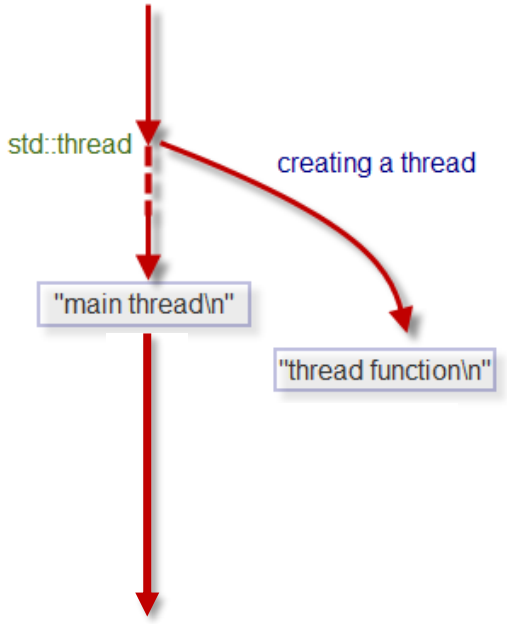


How do we get our application to simultaneously execute different lines of code?

Answer: **Threads!**

Threads allow lines of code to be executed simultaneously

We can organize our code into functions, give the thread the function to execute, and continue executing other code while the thread works at the same time



How do we make a thread in C++?

Step 0: Include standard thread library

```
#include <thread>
```

Step 1: Organize simultaneous behavior into functions

```
// A dummy function
void foo(int Z)
```

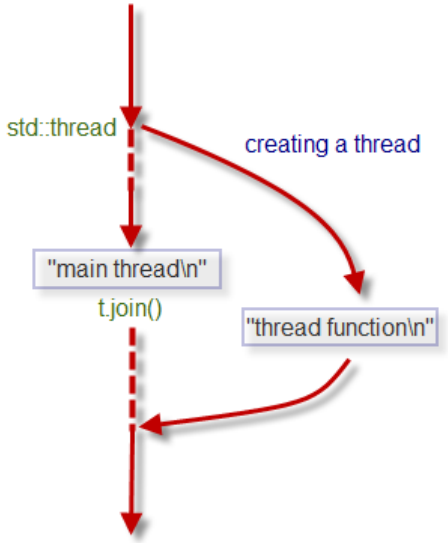
Step 2: Create thread object and give it the function to execute

```
thread th1(foo, 3);
```

What do we do if we need data that a thread is operating on?

We can wait for a thread to finish with the “join” function

The “join” function will stall the code until the thread is finished




```
th1.join();
```

C++ Thread Demo

Let’s thread our image processing application

We’ll functionalize the edge detection algorithm and file access

A thread will then call the function once the user specifies the file

Threaded Application Demo

Success!

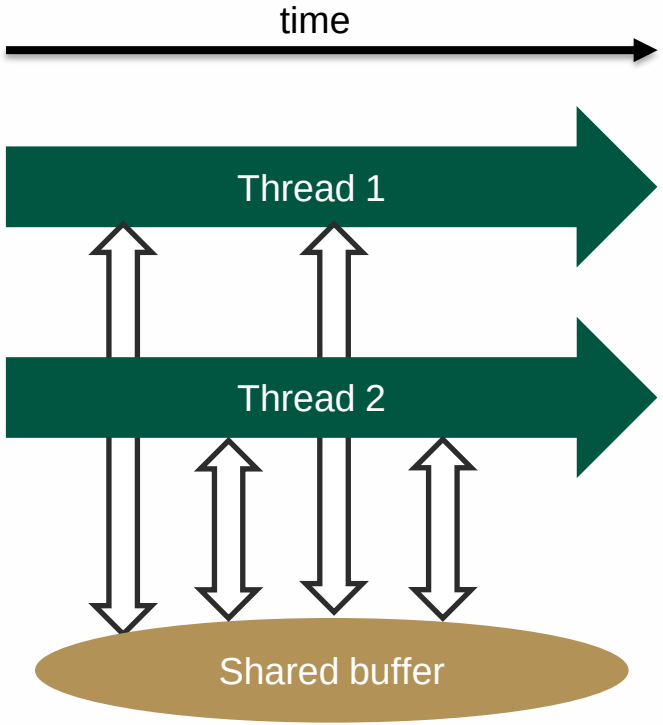
The user input can be collected while at the same time the image is processed!

But wait… what happened to our images?

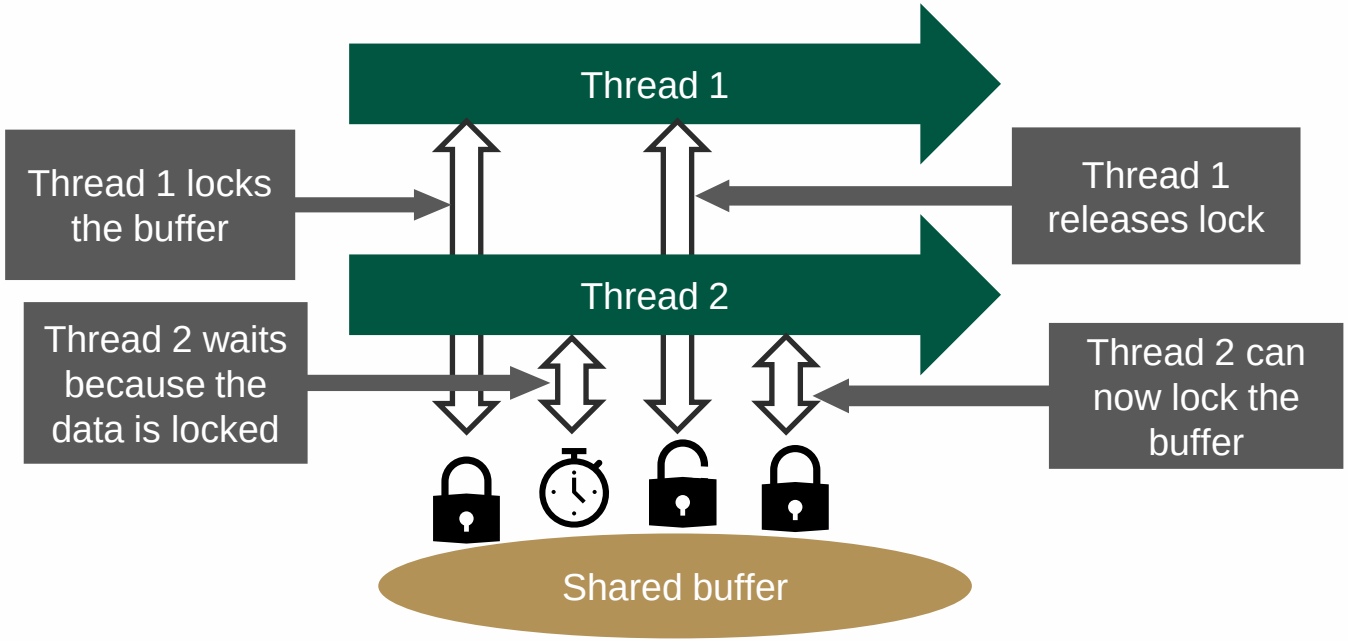
Our image processing functions share a temporary buffer when working on images…

When multiple threads run at the same time, they both use this same buffer and write over each others' data!

This is called a race condition



Solution: We need to lock the data!



This kind of lock is called a mutex (short for **MUT**ual **EX**clusion)

A mutex can be locked by a thread to secure a resource

If another thread tries to lock a mutex that is already locked, it will stall until it can acquire the lock

How do we make a mutex in C++?

Step 0: Include mutex library

```
#include <mutex>
```

Step 1: Create global mutex object

```
mutex m;
```

Step 2: Lock and unlock the mutex to secure the resource

```
m.lock();
m.unlock();
```

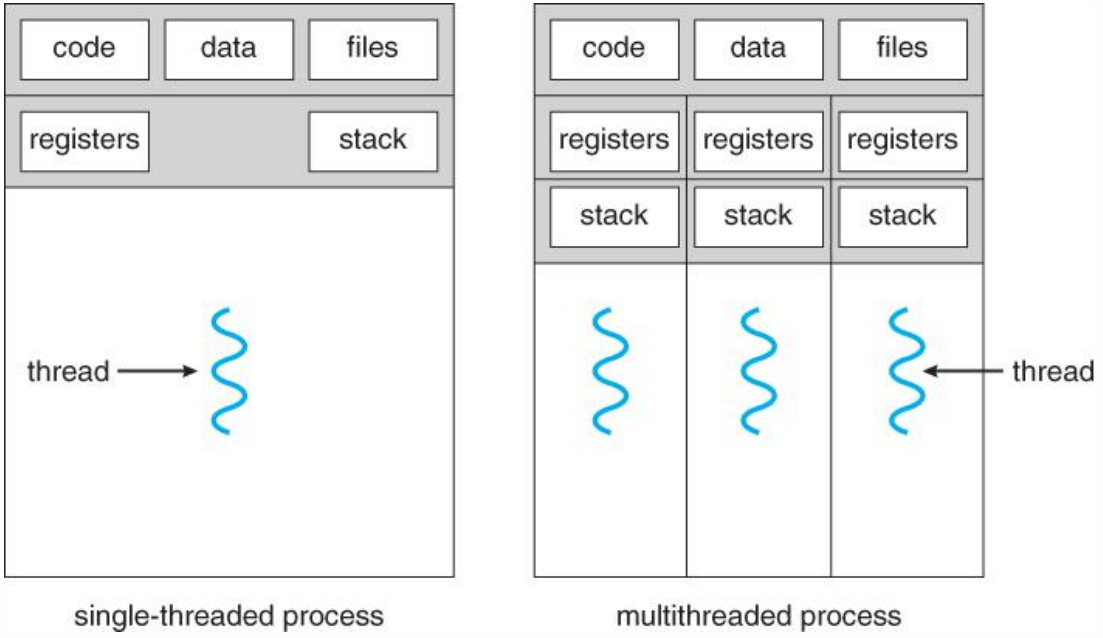
Mutex Locked and Threaded Application Demo

How does the computer simultaneously execute multiple threads?

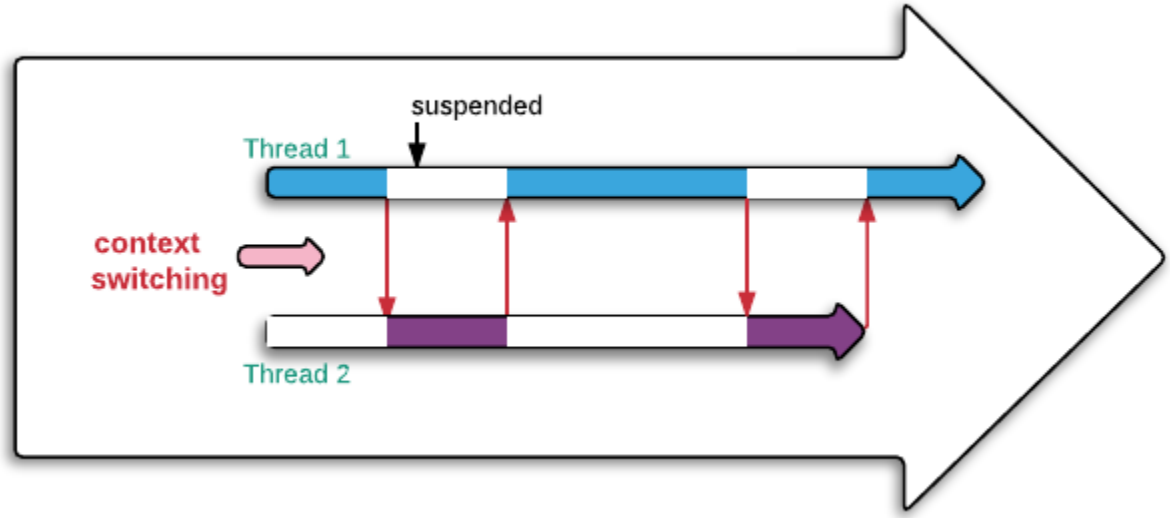
The **operating system** manages memory for each thread and then swaps back and forth between each thread

The operating system creates separate stack space for each thread

Other memory space is shared (heap, data, bss)

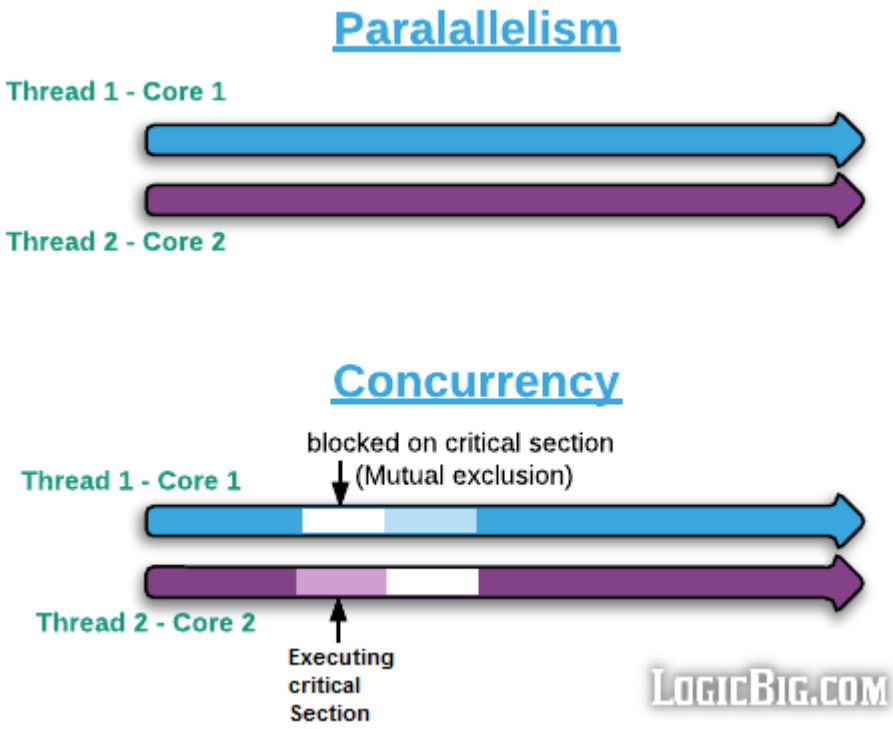


The operating system's thread scheduler lets each thread have execution time on the CPU



A single-core CPU will have to run threads concurrently

Multi-core CPUs can run threads in parallel

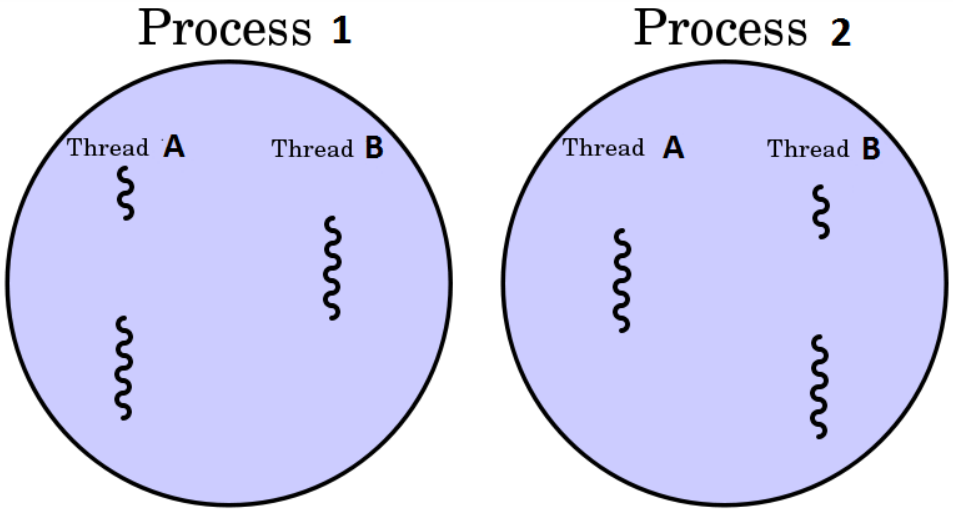


THE WILLIAM STATES LEE COLLEGE OF ENGINEERING

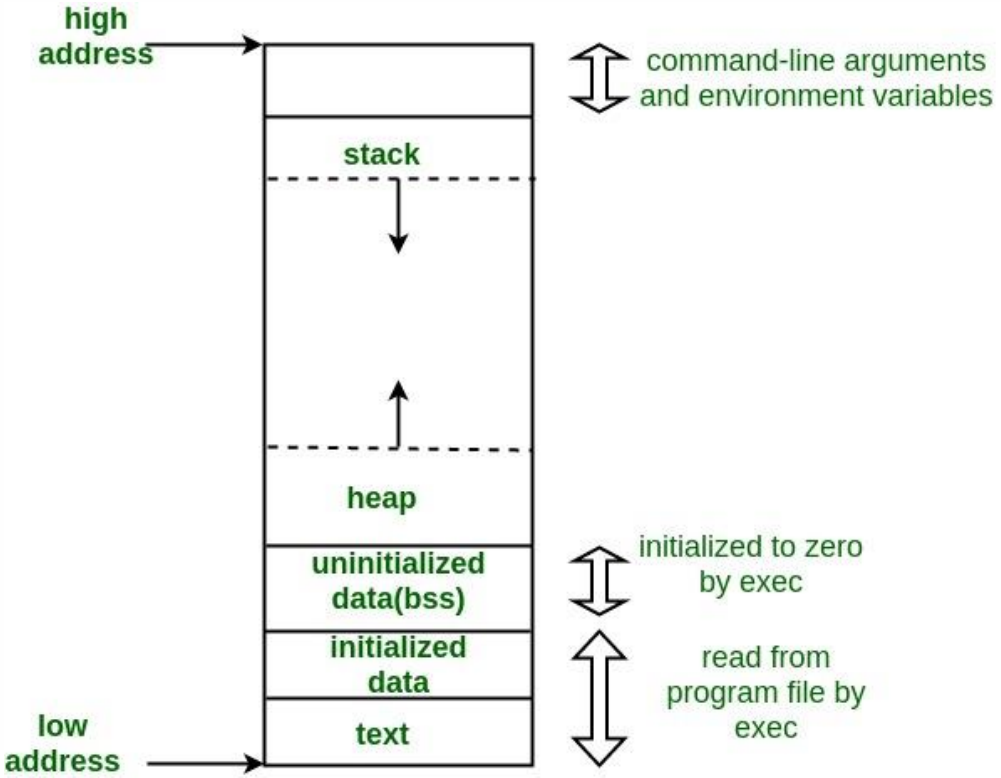
Thank You for Coming!

Questions?

Appendix A:



Appendix B:


high address
stack
heap
uninitialized data(bss)
initialized data
text
low address
command-line arguments and environment variables
initialized to zero by exec
read from program file by exec

Appendix C:

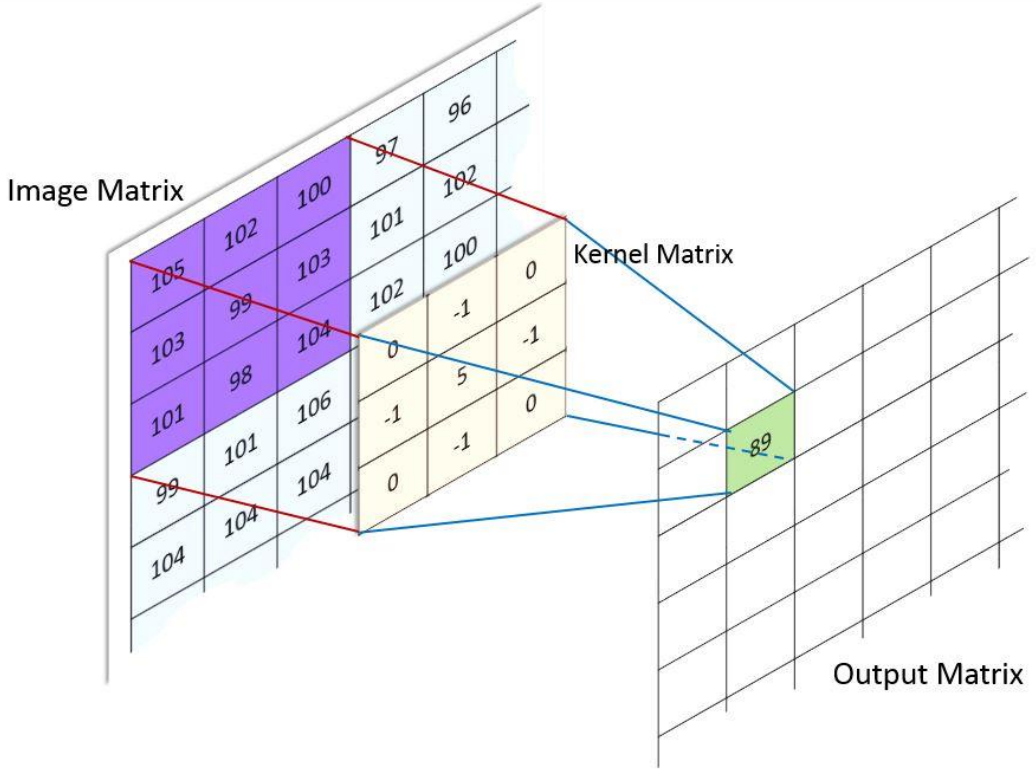
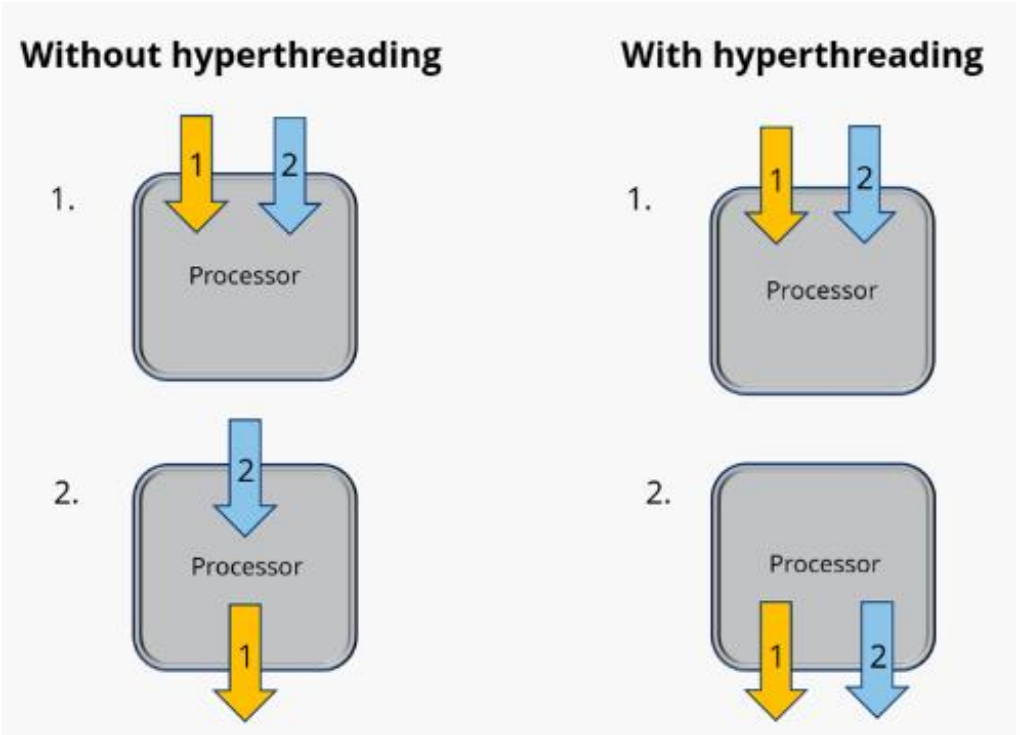
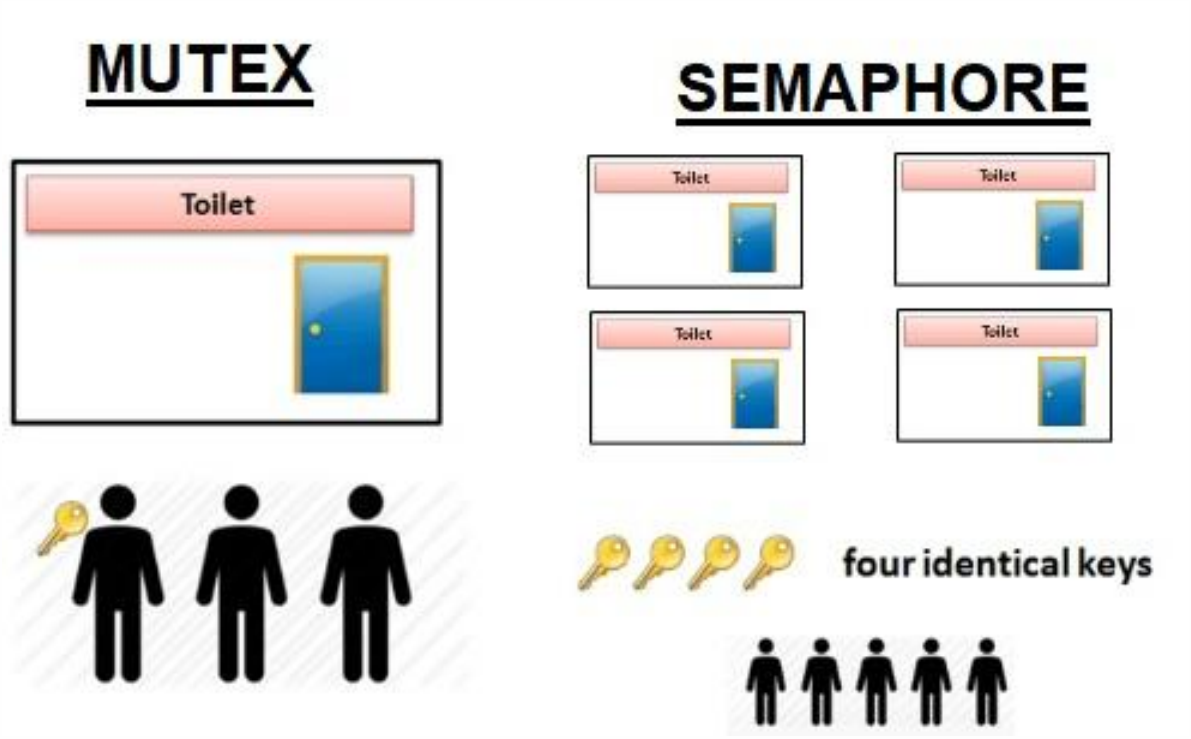


Image Matrix
105
102
100
97
96
103
99
103
101
102
101
98
104
0
-1
-1
99
101
106
-1
5
0
104
104
104
0
-1
-89
Kernel Matrix
Output Matrix

Appendix D:


Without hyperthreading
With hyperthreading
1.
2.
Processor
1
2

Appendix E:


MUTEX
Toilet
SEMAPHORE
Toilet
Toilet
Toilet
Toilet
four identical keys