Differences observed between process created with fork() & pthread_create():

[For both cases the global variable x = 10]

- fork()
    - The child process decrements the global variable linearly from **10 to -90**
    - The Parent process increments the global variable linearly from **10 to 100**

- pthread_create()
    - The parent thread increments the global variable linearly from **10 to 100**
    - The child thread decrements the global variable linearly from **100 to -90**

Reasons for the differences observed

In both cases the parent and child execute concurrently.
However, parent and child processes are independent of one another. Meaning that they don't share memory i.e. they are isolated from one another. Hence, both the processes read the value of the global variable as 10 at the time of execution because their memory and code are independent of each other.

Threads are not independent of one other like processes as a result threads share with other threads their code section, data section and OS resources like open files and signals. Hence, when the parent thread has incremented the global variable to 100, the child thread using the same heap memory (different stack) reads its value as 100 not 10 and then linearly decrements it to -90.

Why does this happen

Inter-process communication is slow as processes have different memory addresses. As an alternative to this, Inter-thread communication can be faster than inter-process communication because threads of the same process share memory with the process they belong to.

Meaning, that if the data has to remain unchanged we can use processes but communication between processes will be slow.
If the data can be changed, then we use threads, because in this case they will run faster.