**Assignment (1.7 Thread and Processing)**

**Brief**

We have learn much about Processes and Threads. By that, you should able to know the key points of Processes and Threads.

* Create a table to differentiate between threads and processes
* You may use any resources available online for research and create your own differentiation of the two concepts.

**Submission**

* Submit the URL of the GitHub Repository that contains your work to NTU black board.

|  |  |  |
| --- | --- | --- |
| **Comparison Basis** | **Process** | **Thread** |
| Definition | A process is a program under execution i.e. an active program. | A thread is a lightweight process that can be managed independently by a scheduler |
| Context switching time | Processes require more time for context switching as they are heavier. | Threads require less time for context switching as they are lighter than processes. |
| Memory Sharing | Processes are totally independent and don’t share memory. | A thread may share some memory with its peer threads. |
| Communication | Communication between processes requires more time than between threads. | Communication between threads requires less time than between processes. |
| Blocked | If a process gets blocked, remaining processes can continue execution. | If a user level thread gets blocked, all of its peer threads also get blocked. |
| Resource Consumption | Processes require more resources than threads. | Threads generally need less resources than processes. |
| Dependency | Individual processes are independent of each other. | Threads are parts of a process and so are dependent. |
| Data and Code sharing | Processes have independent data and code segments. | A thread shares the data segment, code segment, files etc. with its peer threads. |
| Treatment by OS | All the different processes are treated separately by the operating system. | All user level peer threads are treated as a single task by the operating system. |
| Time for creation | Processes require more time for creation. | Threads require less time for creation. |
| Time for termination | Processes require more time for termination. | Threads require less time for termination. |
| Lightweight | The process is not Lightweight | Threads are Lightweight |
| Work and Conception | It takes more time for work and conception | It takes less time for work and conception |
| Multiprogramming | Multiprogramming holds the concepts of multi-process. | We don’t need multi programs in action for multiple threads because a single process consists of multiple threads. |
| Relationship between Parent and child | The process has its own Process Control Block, Stack, and Address Space. Changes to the parent process do not affect child processes. | Thread has Parents’ PCB, its own Thread Control Block, and Stack and common Address space. Since all threads of the same process share address space and other resources so any changes to the main thread may affect the behavior of the other threads of the process. |
| System Call | A system call is involved in it. | No system call is involved, it is created using APIs. |
|  |  |  |