# **Q:- What is the difference between the process and the threads?**

## Ans:- Processs:-

It is an independent program in the execution. It has its own memory space , system resources and execution state.Each process operates in its on address space , isolated from others.

High memory usage for each process.

IPC(inter process communication ) like pipes , sockets and message queues are required to share data and communication. Need safety for communication so it is slow.

Forking (creating a new process) ia an expensive task reuires allocating new memory space and copying result.

Due to isolation , provide higher level of security and stability, if one crashes does not effect all.

Context switching is expensive as due to load and save different memory maps and resources.

Suitable for tasks that require heavy isolation, such as running different applications simultaneously, web servers handling multiple clients, or different services on an operating system.

Examples:- Running a web browser and and word processor

## Threads:-

A thread, often called a lightweight process, is a smaller unit of a process that can be scheduled execution . Multiple threads can be exist within a single process, sharing the same memory space and resources.

Low memory usage as used share memory. But caused the race condition

Straightforward, fast and impler communication and resource sharing as they all share common resources.

Thread creation is less expensive.

A failure in one can potentially corrupt the shared memory , affecting other within the same process.

Context switching between threads is faster since they share the same memory space, and only the registers, program counter, and stack pointer need to be switched.

Ideal for tasks that require concurrent execution within the same application, like handling multiple requests in a web server, performing background tasks in a desktop application, or parallel processing in computational tasks.

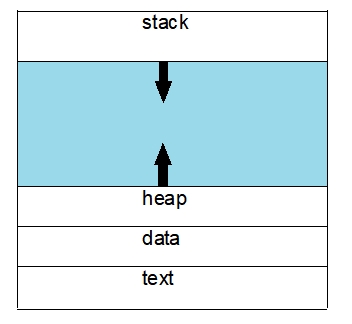
Ideal for tasks that require concurrent execution within the same application, like handling multiple requests in a web server, performing background tasks in a desktop application, or parallel processing in computational tasks.

Example:- A web browser may have multiple tabs open, each running as a separate thread within the same browser process.

# Q:- What is the race condition ?

Q:-3: How the process store in the memory?

Ans:- A program in execution is called process.   
Lets looks at each memory section that how process is look like within memory



Text:-   
This section of the memory contains the executable instructions of a program. It also contains constants, macros. It is read only to prevent accidentally modification of an instruction. Shareable for other processes.

Data:-   
It contains the gloabal and static variables that are intilized by the programmer prior to the execution of a program. The values of the variables can be changed so it is not read\_only part.

Heap:-  
 Dynamic memory allocation is done to allocate the memory for the variables whose size can not be statically determined by the compiler before the program execution and this is requested by the programmer. This is done in heap segment and determined at run time. It is managed via system calls to envoirnment variable (malloc, calloc,) free, delete etc.

Stack:-  
 A process generally also includes the process stack, which contains temporary data i.e. function parameters, return addresses, and local variables. On the standard x86 architecture it grows downwards to lower addresses but on some other architectures it may grow the opposite direction.it is shown in the diagram that stack grows opposite direction of heap for avoiding overlapping problem.