Multitasking, Multithreading & Multiprocessing.

Multitasking
> Doing multiple task at same time.
> ! computer do not multitask.

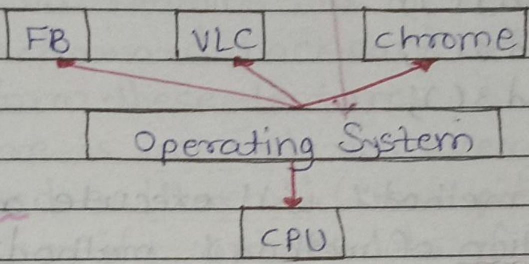
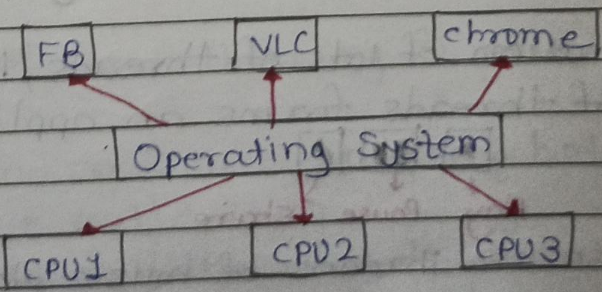


Fig. Multitasking.

Multitasking have two types
1) Multiprocessing (process base multitasking)
2) Multithreading (~~process~~ thread base multitasking)

Multiprocessing
> Here computer uses more than one CPU at a time.
> This increases the speed of processing.



- Single threading
  > Processing/execution happens one by one / step by step.

```
method1();
method2();
method3();
```

> Here first method 1 will execute after full execution of method 1, method 2 will execute and further same.
> If method 2 take 5sec to execute system will wait for 5sec but it will not execute method 3 although it is capable.
> After execution of full method 2 method 3 will get executed.

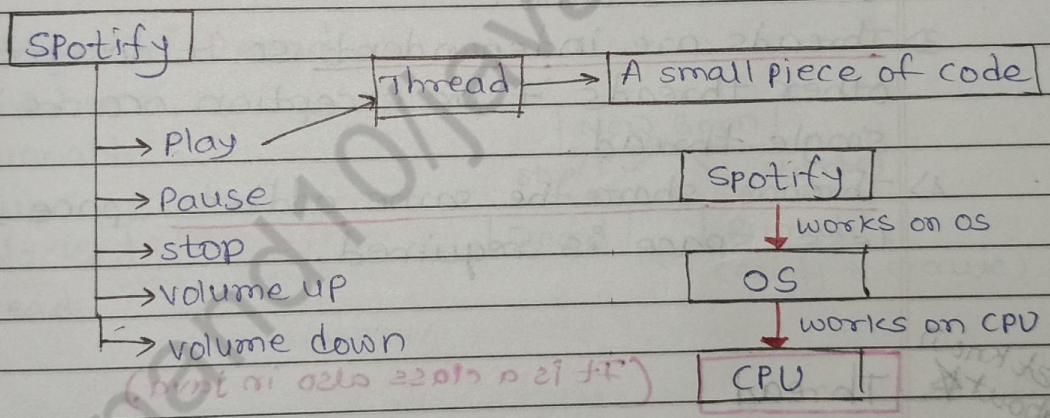
- Let's understand, ~~wa~~ what is thread ?
  > Thread is a lightweight subprocess / small task / piece of code.
  eg. ▶ play button of spotify is one thread.

> Combination of lots of threads / by combing lots of threads forms an application.
Like 

these are threads of music player combining these a whole music player application will form.

Multithreading.

> Process of executing multiple threads simultaneously
> Understand this using a example of spotify
- As VLC contains lots of features like play, pause, stop, forward, backward, volume +-, etc to perform these features.
- There is an code for every task which is called as 'thread'. (small piece of code/program/task)
- As spotify contains lots of different threads. So these threads work at single time due to this we can use whole spotify application.



Diff. betⁿ multithreading & multiprocessing.

> Multithreading & multiprocessing both are used to achieve multitasking.
> We can use multithreading than multiprocessing because threads use a shared memory area. & No separate memory area is allocated to threads so they saves memory.
> switching between threads takes less time than

Takes more time compir to threads.

to switch between processes.

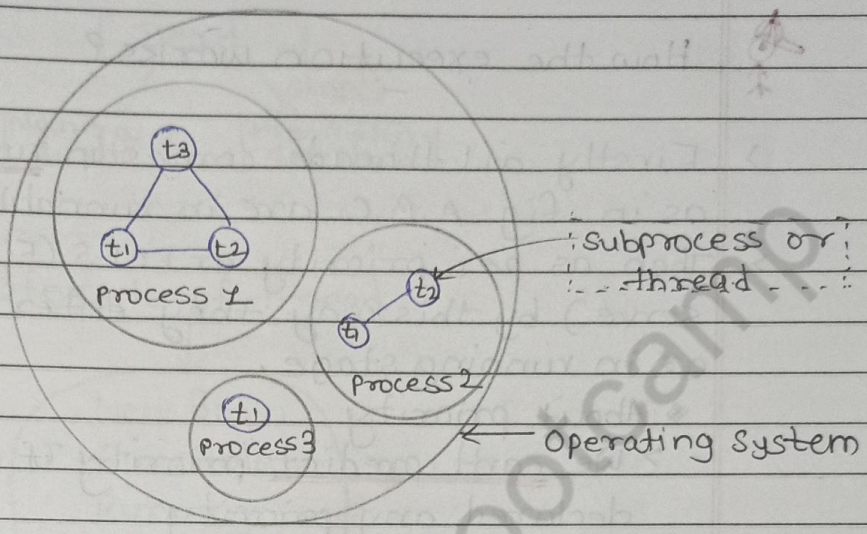
> Java multiprocessing is used in and java multithreading is mostly used in games, animation, etc.

⊙ Advantages of multithreading.

1) Dosen't blocks the user because threads are independent and you can perform multiple operations at the same time.
2) You can perform many operations together so it saves time.
3) Threads are independent, so it dosen't affect other threads if an exception occurs in single thread.
4) Threads share the same address space, so less space is required.

Let know about ✱ **Thread** (It is a class also in java)

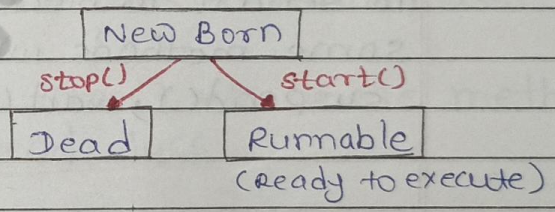
> A thread is a lightweight subprocess, it is the smallest unit of processing. It is a separate path of execution.
> Like in spotify app play, pause are threads.
> A thread is executed inside the process.
> There is context switching between threads.
> There can be multiple processes inside the os.
> One process can have multiple threads.
> At one time only one thread get executed.

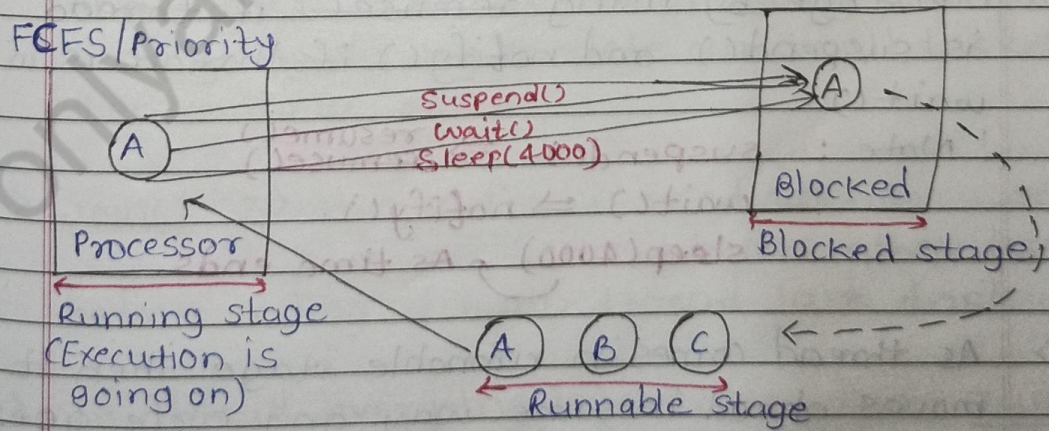


✱ Life cycle of thread.

Stages of thread.

1) New born
2) Runnable
3) Running
4) Blocked
5) Dead





How the execution works?

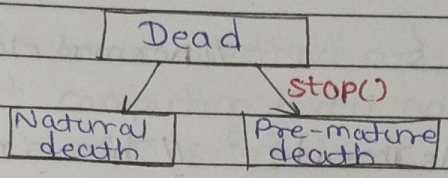
> Firstly all threads comes in runnable stage as in fig A,B,C are in runnable stage.
> Then as per priority or FCFS (First come first serve) by this way they enter in processor or in running stage.
  - About priority
    > We can't predict priority if we haven't declared any priority
    > By default priority of thread is 5 and max it can be 10

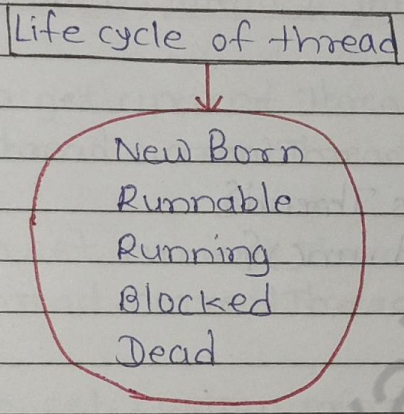
> As thread starts executing and if we want to send it to blocked stage there are some methods which can be used like suspend(), wait(), sleep(4000)
  ↳ This method comes with timer. After time get completed the thread comes in runnable stage.

> We can get blocked threads again in runnable stage by using ~~remuse()~~ resume() if thread is suspend() and notify() if thread is in wait()

! Note: suspend() ⇒ ~~remuse()~~ resume()
wait() ⇒ notify()
sleep(4000) ⇒ As time ends.

> As thread comes in runnable stage then it moves ahead in same way and cycle goes on.





* Let's understand some methods in Thread class.

> yeld(), wait(), nofify(), stop(), etc these methods are included in Thread class.

> yeld() ⇒ This is used in case the process not stoping but we want to complete that task for that we are going to request and going to say complete our small task & then continue.

> join() ⇒ This method is used when we want to end specific task before other task or before ending of main thread last task.

> In thread everyting is executed from run();

Multithreading example:-

```
class A{.....}          //normal class

class B extends Thread{     //This is now extending thread class
    public void run(){      //This method is used to perform action for thread.
        //code
    }
}

class C extends Thread{
    public void run(){
        //code
    }
}

public class Main{
    public static void main(String[] args){
        A ao = new A();
        B bo = new B();
        C co = new C();

        bo.start();  //This will start run() in class B
        co.start();

        try{bo.join();      //This will let whole thread execute and after that let parent thread further method execute.
            co.join();}
            catch(Exception e){}
            System.out.print("The end!");
    }
}
```

join() throws exception so we have to handle it

O/P: // execution of threads. whose sequence can't be predicted.
The end!

> JVM treats main method as parent thread.
> By default computer given name to main method or main thread is 'main'.
> If there are three threads then their names will Thread-0, Thread-1, Thread-2 ....

> Method to get current Thread Name
```
Thread.currentThread().getName();
```

> Method to set current Thread Name.
```
Thread.currentThread().setName("customNm");
```

> Method to set custom priority
```
A a0 = new A();
a0.setPriority(8);
```
⊙ priority ranges from 1 to 10
⊙ - Default priority is 5 (NORM-PRIORITY)
- MIN-PRIORITY is 1
- MAX-PRIORITY is 10

! More the priority more the time thread will get to execute.

"Consider"
> Like we have two threads 'A' and 'B'
> Thread 'A' have priority '8' and 'B' have priority '2'
> If program have 2 sec to execute then from those 2 sec thread A will be getting more time compair to thread B
> Understand it like thread A will print numbers from 1 to 100 while thread B will print numbers from 1 to 20 only.

! Priority in thread does not work always. Because multithreading is not enabled in most of computer systems it need to be enabled which is possible.

> method used to check that thread is alive or not ao.isAlive(); //Gives boolean value

> As thread get completly executed it will get automatically dead.

* Synchronization

→ ~~In multiph~~

> In multithreading we can't predict what will execute first so synchronization means after execution (full execution) of method by first thread then only second thread will get able to executed that method.

> This is achieved by using synchronized keyword in method this will block another method thread to execute same method till first thread execute it fully.

e.g. public synchronized void method(){...}