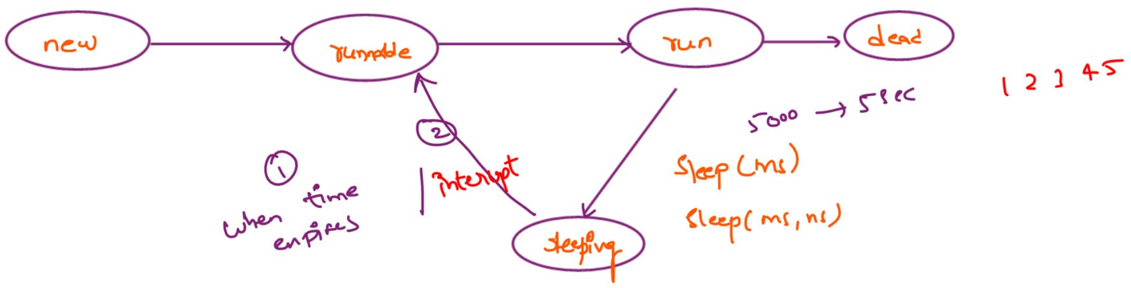
Sleep



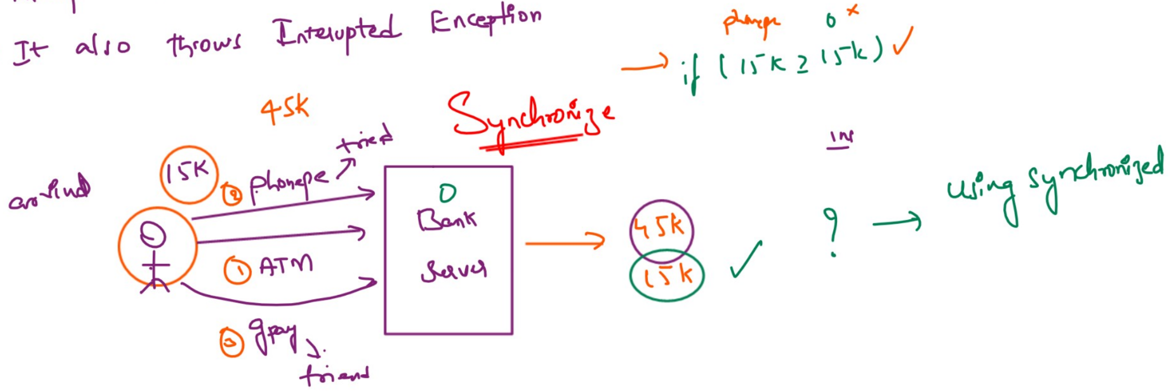
Notes :

→ when we call the sleep( ), the current executing thread, will go to sleeping state till the time specified

→ sleep( ) throws InteruptedException, it is our duty as a developer to handle the exception using try / catch

Interupt( ) : when interupt( ) method is called the thread in waiting state / sleeping state will be interupted, when interupt( ) is called

→ It also throws Interupted Exception



Implementation: → modified

$\boxed{\text{synchronized}}$ method ( )

Note:

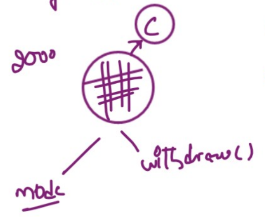
→ In the program scenario, the customer is trying to withdraw the amount from three different modes at the same time, this should not be allowed as it might gives loss to the bank. [ie gives inconsistent output]

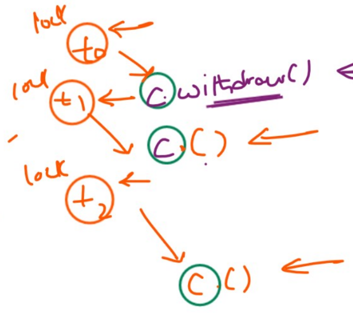
→ We need to allow only one mode [thread] to withdraw the money at a time

→ To achieve this we can make use of synchronized keyword for the method which has to be synchronized the threads

→ synchronized is a modifier in java

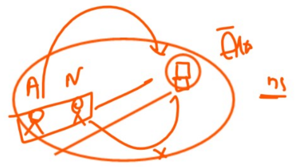
→ when Synchronized keyword is used in a instance method, the object [Customer instance i.e arvind] accessing that method would get locked. ie the object that is accessing synchronized method won't be able to access any other synchronized methods of that class





← executed (run state) → dead/stop → unlock

won't be allowed to run/call the synchronized withdraw() because it is locked



applied in 2 levels

static method	—	class level
instance method	—	object level

Book My Show
RedBus
↑
It locks all the objects of that class

It locks the particular object (single)

<u>Conclusion:</u>

Since the same customer object was accessing the withdraw() method through 3 different threads, there we have to lock the object customer