Q1/ What is the difference between a synchronized method and a synchronized block?

Synchronized Method	Synchronized Block
● It will provide lock on either object level or class level. ● The scope of the synchronized method is on entire functionally, comparatively has greater scope. ● Performance of the synchronized method is low. ● Waiting time of the synchronized method is high. ● It will not throw nullPointerException.	● It will provide lock on any object (specified in parameter). ● The scope is limited only to some statement, comparatively has lesser scope. ● Performance of the synchronized block in high. ● Waiting time of the synchronized block is low. ● It will throw nullPointerException.

Q2/- What's the difference between class lock and object lock?

Object Level Lock	Class Level Lock
● It can be used when you want non-static method or non-static block of the code should be accessed by only one thread ● It should always be used to make non-static data thread safe. ● Every object the class may have their own lock	● It can be used when we want to prevent multiple threads to enter the synchronized block in any of all available instances on runtime ● It should always be used to make static data thread safe. ● Multiple objects of class may exist but there is always one class's class object lock available

Q4/- What is a Race condition, How will you solve the Race condition,explain with an Example.
Java is a multi-threaded programming language and there is a higher risk to occur race conditions. Because the same resource may be accessed by multiple threads at the same time and may change the data.

A race-condition is a condition in which the critical section (a part of the program where shared memory is accessed) is concurrently executed by two or more threads. It leads to incorrect behavior of a program.

Race condition can be solved by synchronizing the process.
Example :

```
class Common{
        public void fun1(Stirng name){
        System.out.print("Welcome");
        try{
            Thread.sleep(1000);
        }catch(Exception ee){
      } System.out.println(name);
}

class ThreadA extends Thread{
          Common c;
          String name;
          public ThreadA(Common c,String name) {
                this.c=c; this.name=name;
         }

        @Override
         public void run() {
              c.fun1(name);
         }
   }

 class ThreadB extends Thread{
```

```
        Common c;
        String name;
        public ThreadB(Common c,String name) {
            this.c=c; this.name=name;
        }

        @Override
        public void run() {
            c.fun1(name);
        }
 }

class Main{
        public static void main(String[] args){
                Common c=new Common();
                //sharing same Common object to two thread
                ThreadA t1=new ThreadA(c,"Ram");
                ThreadB t2=new ThreadB(c,"Shyam");

                t1.start();
                t2.start();
          }
}
```

Q5/- What is the Difference between the sleep and join method .

Join()	**Sleep()**
● If a thread wants to wait until completing thread some other threads, then we should go for join. ● It is not a static method. ● It is a final method.	● If a thread doesn't want to perform any operation for a particular amount of time, then we should go for sleep() method. ● It is a static method. ● It is not a final.

• It will make to wait for other thread to complete.	• It will slow down the thread for specific period of time.