**Concurrent Collection– Part\_01**

* **Need for Concurrent Collection:**

1. Most of the normal collection classes are not thread safe, so there will be a data inconsistency.
2. There are few Collections (Vector, Hashtable, synchronizedList(), synchronizedSet(), synchronizedMap(), which has thread-safety, but only one thread can access those collections at a time, this increases the waiting time of thread, which in-turn causes performance issue.
3. When one thread reads a value from collection, in the meantime another thread tires to update the value of the collection, then the iterating collection fails with runtime exception saying: ConcurrentModificationException

* **Program for ConcurrentModificationException:**

import java.util.\*;

class MyThread extends Thread{

static ArrayList l = new ArrayList();

public void run(){

try{

Thread.sleep(2000);

} catch(InterruptedException ie){

}

l.add(“D”);

}

public static void main(String[] args) throws InterruptedException{

l.add(“A”);

l.add(“B”);

l.add(“C”);

MyThread t = new MyThread();

t.start();

Iterator itr = l.iterator();

while(itr.hasNext()){

String s1 = (String) itr.next();

System.out.println(“Main Thread Iterating List and Current object is:”+s1);

Thread.sleep(3000);

}

System.out.println(l);

}

}

**Output:**

Main Thread Iterating List and Current object is: A

Child Thread updating list

Exception in thread “main” java.util.ConcurrentModificationException