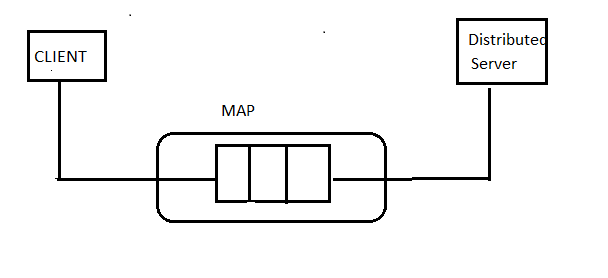
Classes:

Randomizer

PrimeChecker

Randimizer is act as a Producer where the Prime is the receiver with the help of distributed queue.



Considering the above diagram Randomizer is Thread1& Prime is the Thread2.

Randomizer has a method randomGenerate which is responsible for generate the random integers& put into the distributedQueue.

package com.prime;

import java.rmi.Remote;

import java.rmi.RemoteException;

import java.util.List;

import java.util.Map;

/\*\*

\* @author KIRANMAI

\*

\*/

public interface PrimeChecker extends Remote {

public Map<Integer, Boolean> primeChecker(List<Integer> randomList) throws RemoteException;

}

Prime has a method primeChecker which is responsible for receiving the distributedQueue with Integers& returning the distributedQueue with list if entities “PrimaryQueue<Value>”.

PrimeCheckerImpl() throws RemoteException{

super();

}

public Map<Integer, Boolean> primeChecker(List<Integer> randList) throws RemoteException{

int i,m=0,flag=0;

Map<Integer,Boolean> primeCheckMap=new HashMap<>();

Iterator<Integer> itrList= randList.iterator();

if(itrList.hasNext()){

int randNum=itrList.next();

m=randNum/2;

for(i=2;i<=m;i++){

if(randNum%i==0){

System.out.println("Number is not prime");

flag=1;

primeCheckMap.put(randNum, false);

break;

}

}

if(flag==0)

System.out.println("Number is prime");

primeCheckMap.put(randNum, true);

}

return primeCheckMap;

}

Passing the Distributed Queue from function randomGenerate() to Prime; common data structure between two threads, a communication channel established.

Public static void main(String[] args) {

Randomizer random = new Randomizer();

Queue<Value> outputQueue = new Prime().primeCheck(random.randomGenerate());

System.out.println(outputQueue);

}

Sample Output:

110873457