NAME: Soham Saha

SECTION: CSE2A

CLASS ROLL: 61

ENROLLMENT NUMBER: 12019009001389

ASSINGEMENT DATE: 28th APRIL,2021

SUBJECT: OOP’S JAVA

Q1)

public class question1 extends Thread{

char ch;

public question1(char a)

{

ch=a;

}

public void run()

{

for(int i=1;i<=5;i++)

{

System.out.println(ch+"->"+i);

}

}

public static void main(String[] args) {

question1 obj1 = new question1('A');

question1 obj2 = new question1('B');

question1 obj3 = new question1('C');

question1 obj4 = new question1('D');

obj1.setPriority(2);

obj2.setPriority(4);

obj3.setPriority(6);

obj4.setPriority(9);

obj1.start();

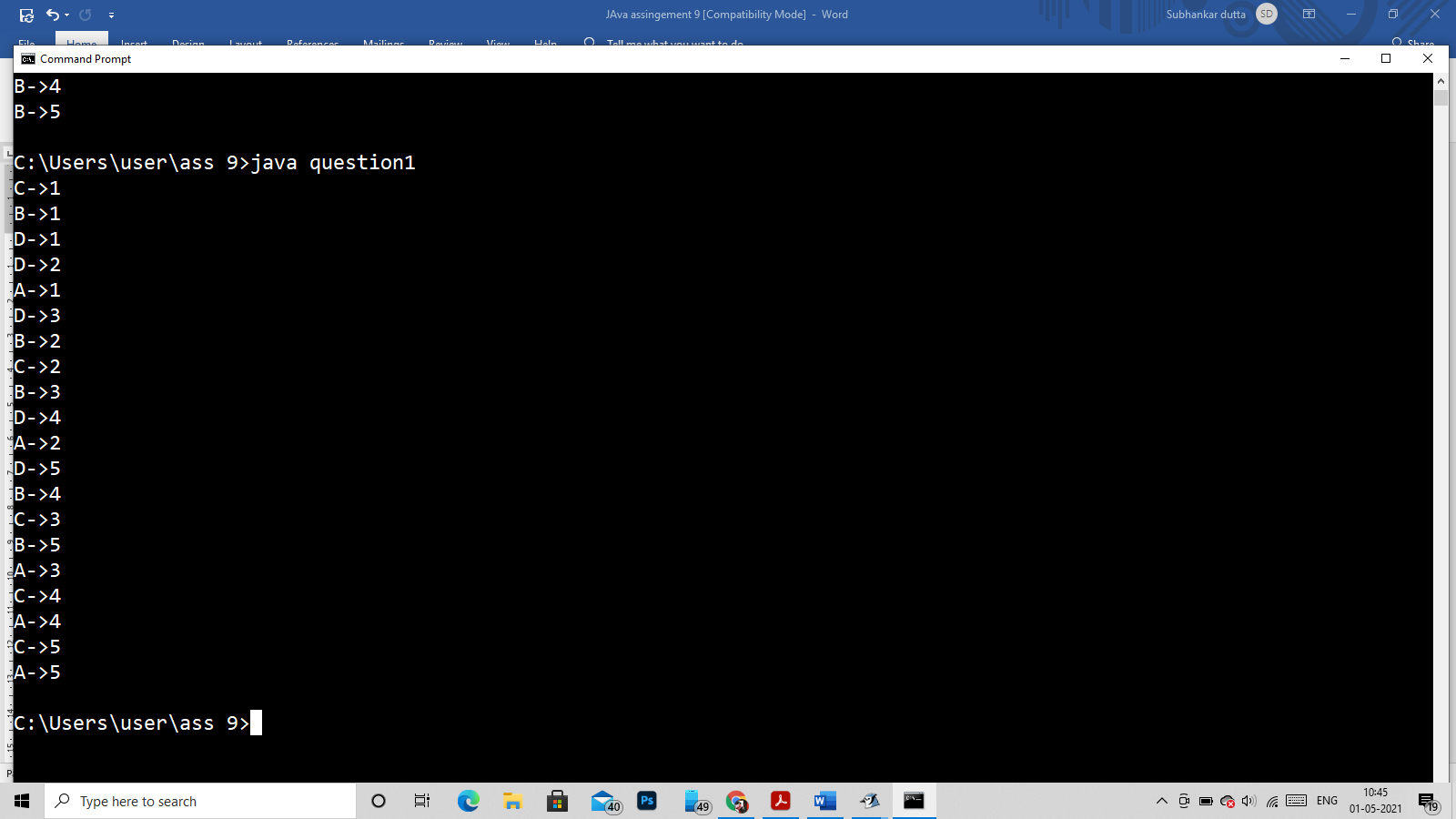
obj2.start();

obj3.start();

obj4.start();

}

}



Q2)

class Test

{

char a;

Test(char a)

{

this.a=a;

}

public void printl(int k)

{

System.out.println(a+ "===Thread started===");

for(int i = 1;i<=5;i++)

{

System.out.println(a+"--->"+k\*i);

}

System.out.println(a+ "===Thread End===");

}

}

class A extends Thread

{

public void run()

{

new Test('A').printl(5);

}

}

class B extends Thread

{

public void run()

{

new Test('B').printl(100);

}

}

class C extends Thread

{

public void run()

{

new Test('C').printl(50);

}

}

class D extends Thread

{

public void run()

{

new Test('D').printl(30);

}

}

public class checkSync

{

public static void main(String asdf[])

{

A a = new A();

B b = new B();

C c = new C();

D d = new D();

a.setPriority(1);

b.setPriority(3);

c.setPriority(5);

d.setPriority(7);

a.start();

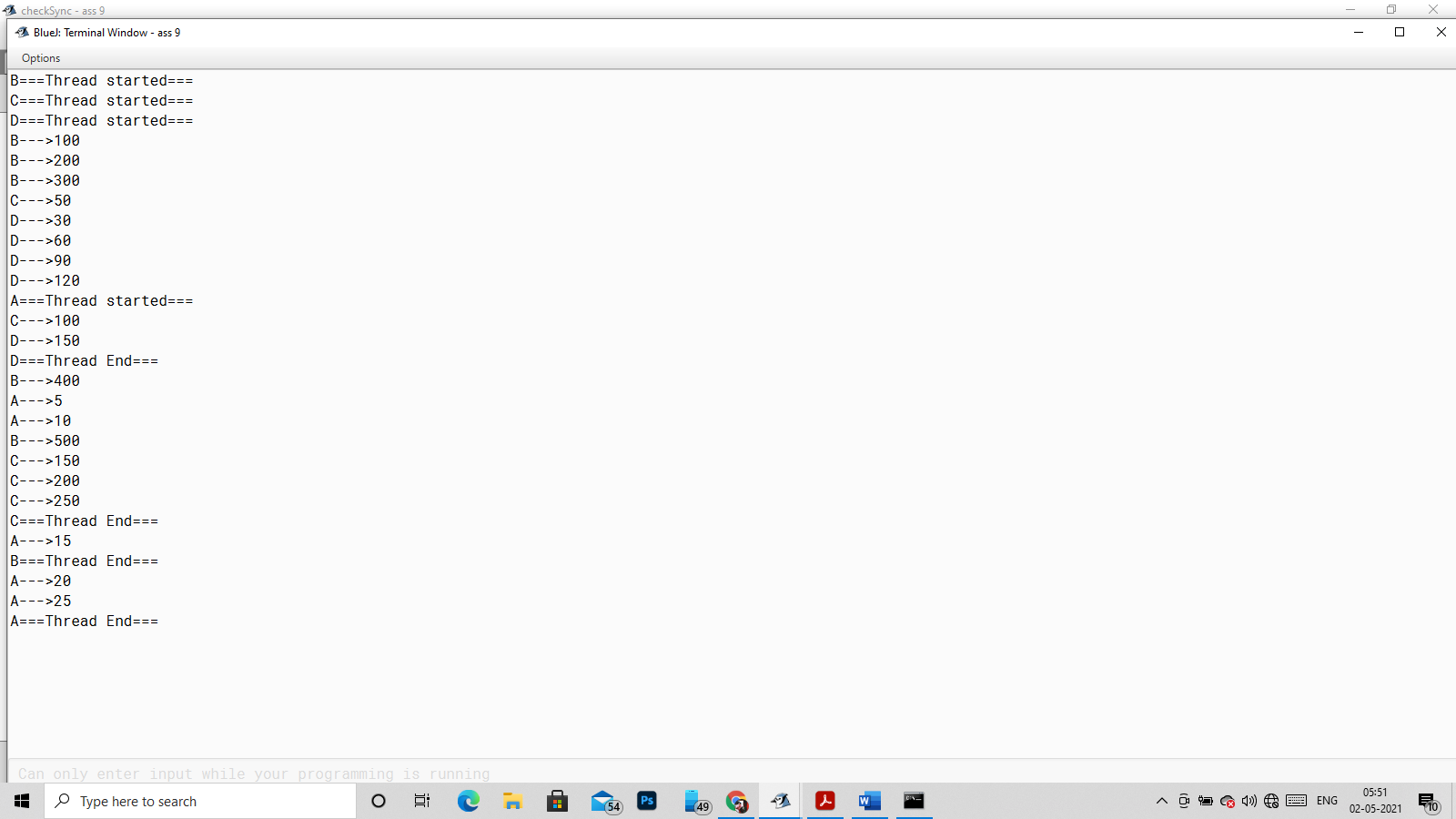
b.start();

c.start();

d.start();

}

}



Q3)

class qes

{

synchronized void print( char ch)

{

for(int i=1;i<=5;i++)

{

System.out.println(ch+"->"+i);

try

{

Thread.sleep(10);

}catch(Exception e)

{

System.out.println(e);

}

}

}

}

class A extends Thread

{

qes t1;

A(qes t1)

{

this.t1=t1;

}

public void run()

{

t1.print('A');

}

}

class B extends Thread

{

qes t1;

B(qes t1)

{

this.t1=t1;

}

public void run()

{

t1.print('B');

}

}

public class question1 extends Thread{

public static void main(String[] args) {

qes q = new qes();

A a = new A(q);

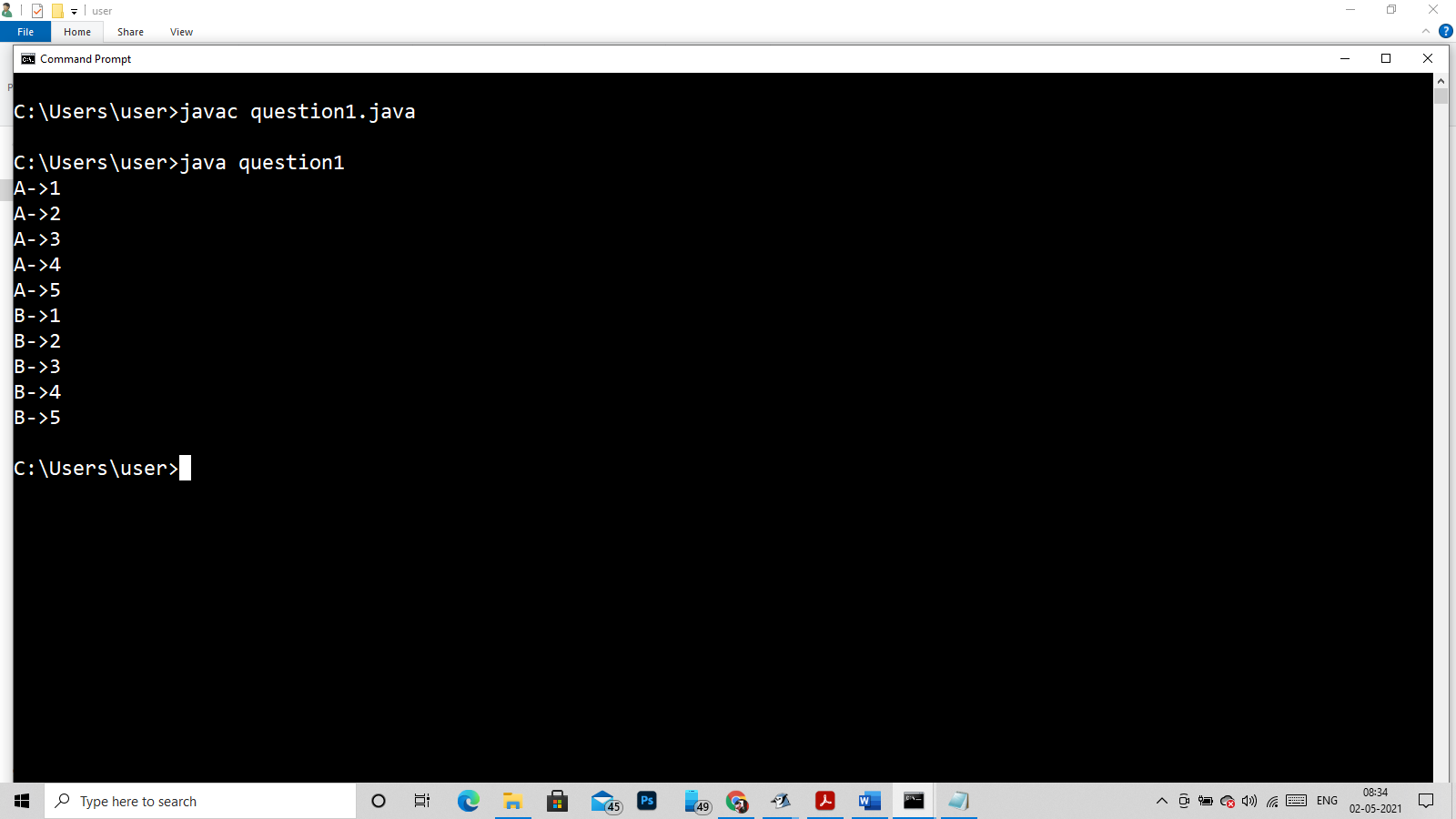
B b = new B(q);

a.run();

b.run();

}

}



Q4)

class qes

{

void print( char ch)

{

synchronized(this)

{

for(int i=1;i<=5;i++)

{

System.out.println(ch+"->"+i);

try

{

Thread.sleep(10);

}catch(Exception e)

{

System.out.println(e);

}

}

}

}

}

class A extends Thread

{

qes t1;

A(qes t1)

{

this.t1=t1;

}

public void run()

{

t1.print('A');

}

}

class B extends Thread

{

qes t1;

B(qes t1)

{

this.t1=t1;

}

public void run()

{

t1.print('B');

}

}

public class question1{

public static void main(String[] args) {

qes q = new qes();

A a = new A(q);

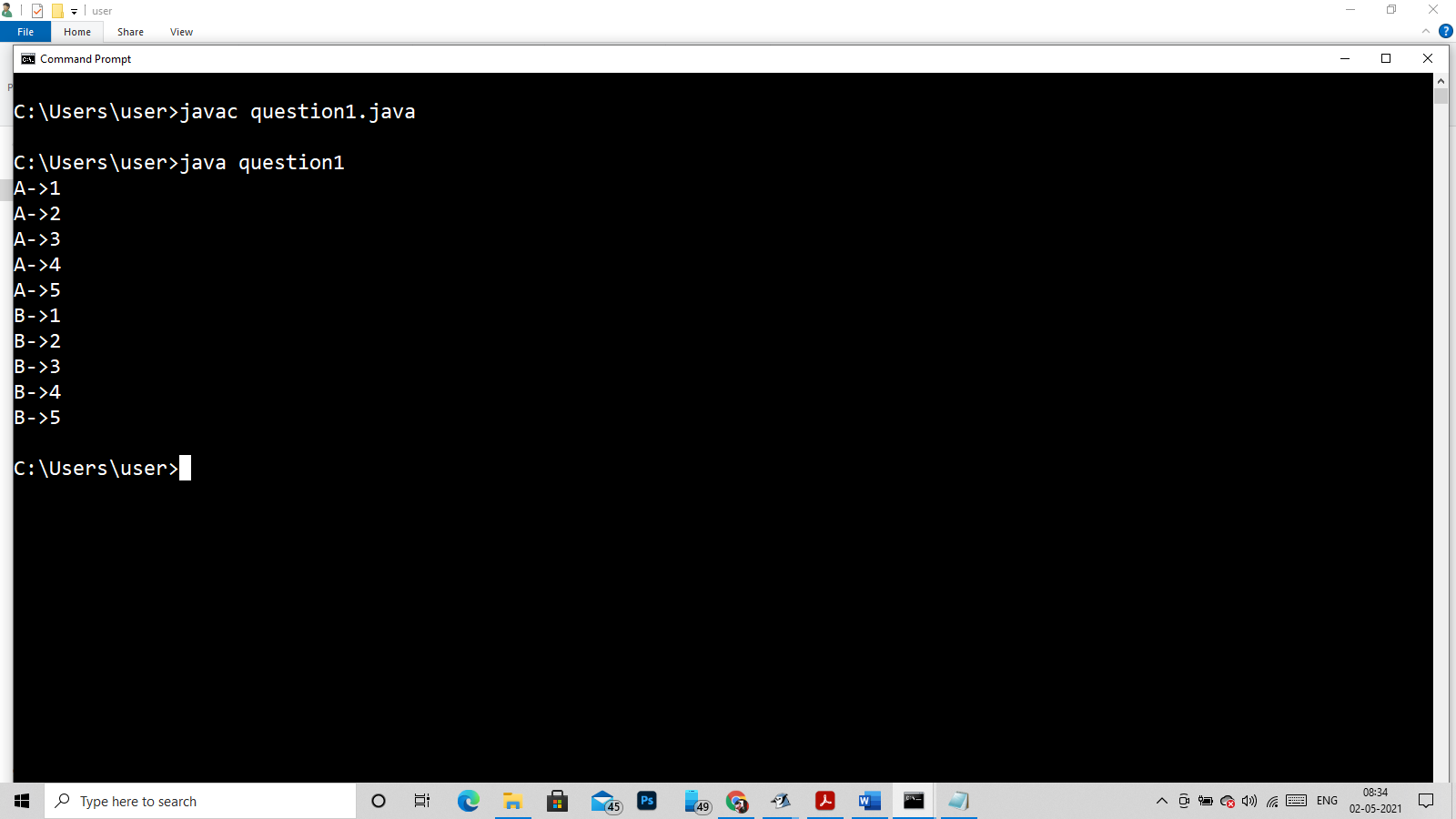
B b = new B(q);

a.run();

b.run();

}

}



5)

public class question1 implements Runnable{

char ch;

question1(char ch)

{

this.ch = ch;

}

public synchronized void run()

{

System.out.println(Thread.currentThread().getName() + " is starting");

for(int i =1;i<=5;i++)

{

System.out.println(Thread.currentThread().getName()+"-->"+i);

try{

Thread.sleep(1000);

}catch(Exception e)

{

System.out.println(e);

}

}

System.out.println(Thread.currentThread().getName() + " is finished");

}

public static void main(String[] args) {

question1 a = new question1('A');

Thread t1 = new Thread(a);

Thread t2 = new Thread(a);

Thread t3 = new Thread(a);

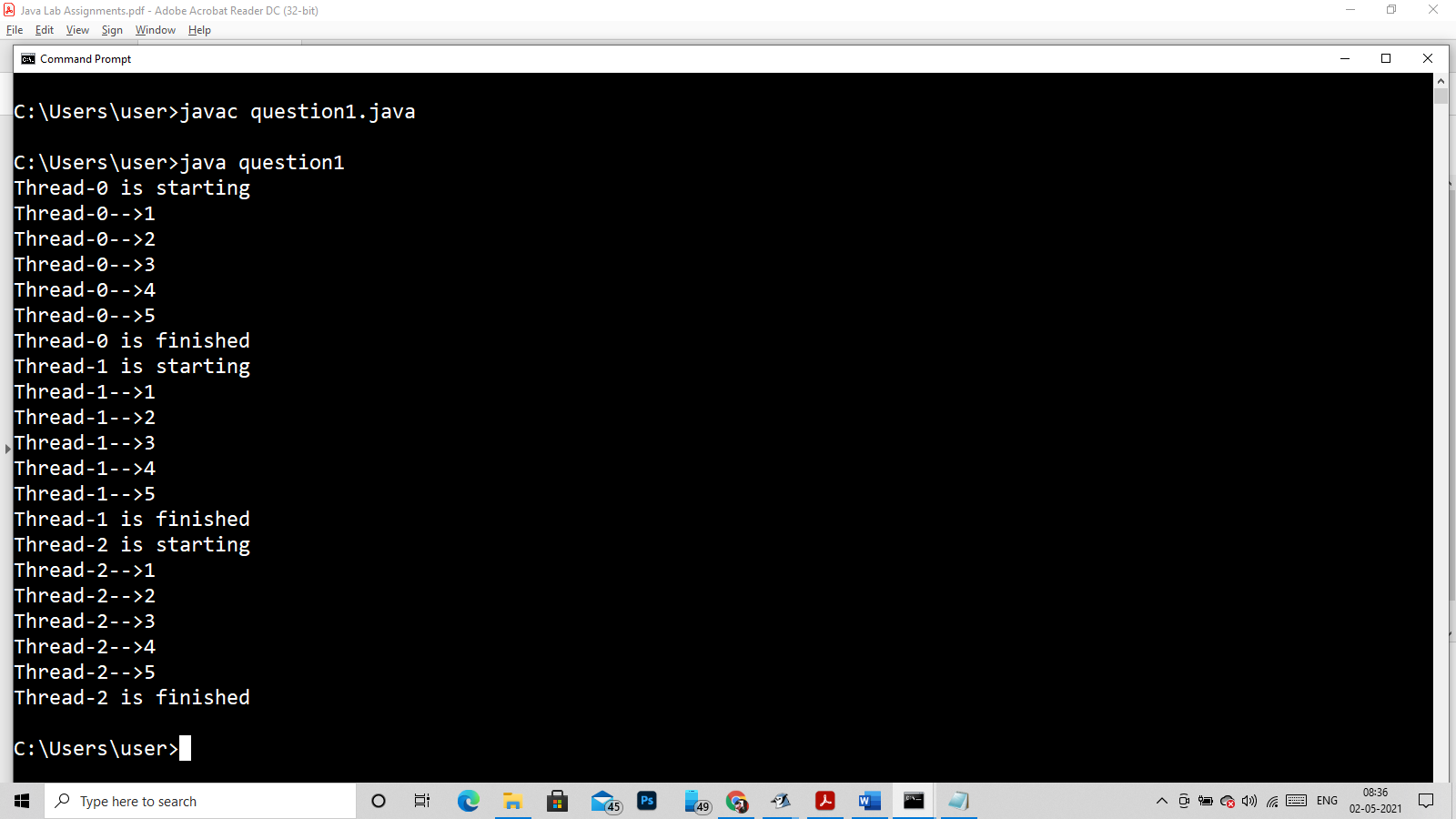
t1.start();

t2.start();

t3.start();

}

}



Q6)

import java.util.\*;

class maker extends Thread

{

int c=3;

LinkedList<Integer> link = new LinkedList<>();

public void producer()

{

int val=0;

while(true)

{

synchronized(this)

{

try

{

//System.out.println("=========== Starting Producing =================");

while(link.size()==c)

wait();

link.add(val);

System.out.println("Producer produce--->"+val);

val++;

notify();

Thread.sleep(500);

// System.out.println("=========== Finishing Consuming =================");

}catch(Exception e)

{

System.out.println(e);

}

}

}

}

public void consumer()

{

while(true)

{

synchronized(this)

{

try

{

//System.out.println("=========== Starting Consuming =================");

while(link.size()==0)

wait();

int val =link.removeLast();

System.out.println("Consumed--> "+val);

notify();

Thread.sleep(500);

// System.out.println("=========== Finishing Consuming =================");

}catch(Exception e)

{

System.out.println(e);

}

}

}

}

}

public class Main

{

public static void main(String args[])throws Exception

{

maker obj = new maker();

Thread t1 = new Thread(new Runnable(){

@Override

public void run()

{

obj.producer();

}

});

Thread t2 = new Thread(new Runnable(){

@Override

public void run()

{

obj.consumer();

}

});

t1.start();

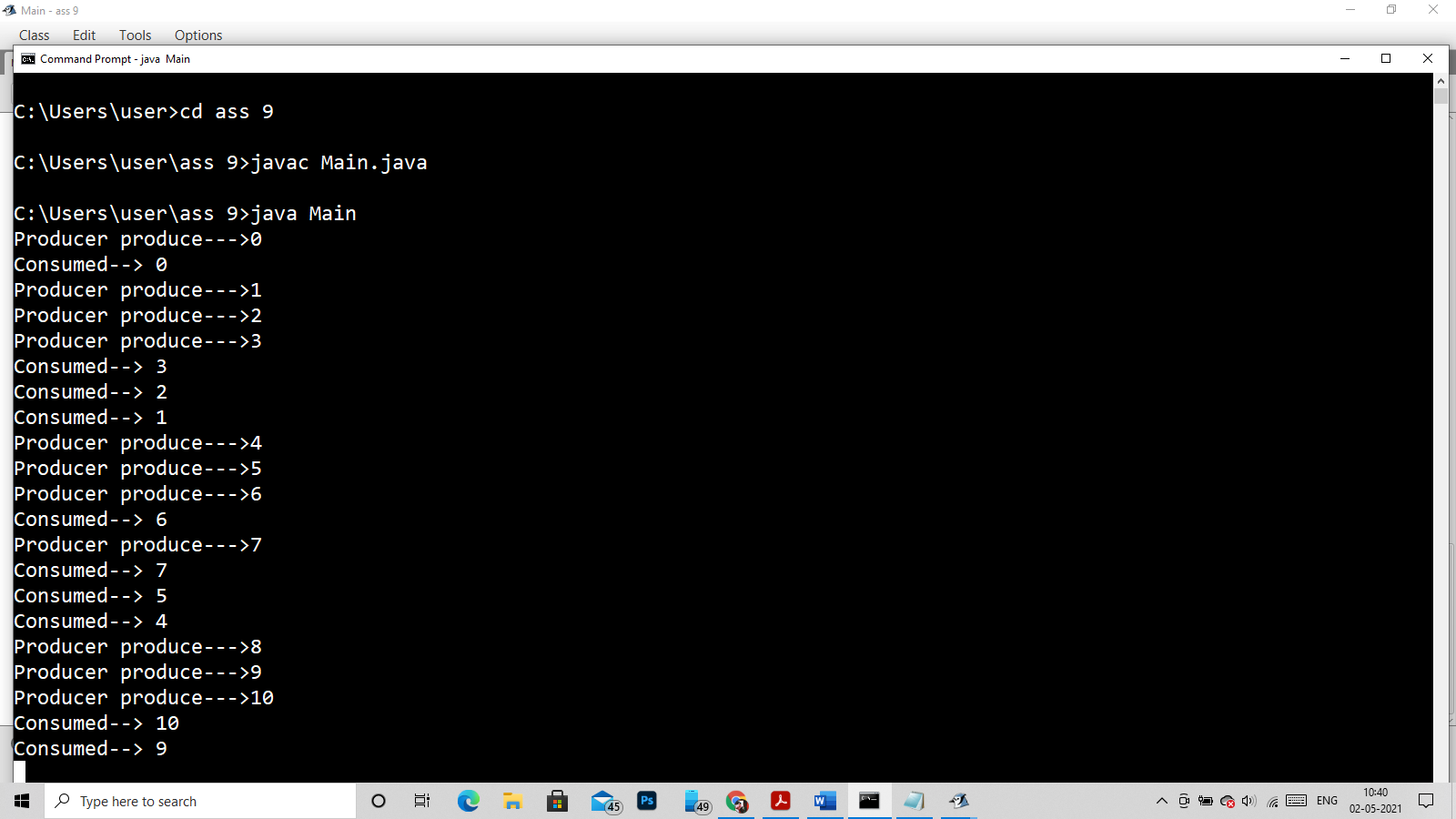
t2.start();

t1.join();

t2.join();

}

}



Q8)

class clasobj implements Runnable {

public void run() {

Lock();

}

public void Lock()

{

System.out.println(Thread.currentThread().getName());

synchronized (this)

{

System.out.println("in block "+ Thread.currentThread().getName());

System.out.println("in block "+Thread.currentThread().getName()+ " end");

}

}

public static void main(String[] args)

{

clasobj c1 = new clasobj();

Thread t1 = new Thread(c1);

Thread t2 = new Thread(c1);

clasobj g2 = new clasobj();

Thread t3 = new Thread(g2);

t1.setName("t1");

t2.setName("t2");

t3.setName("t3");

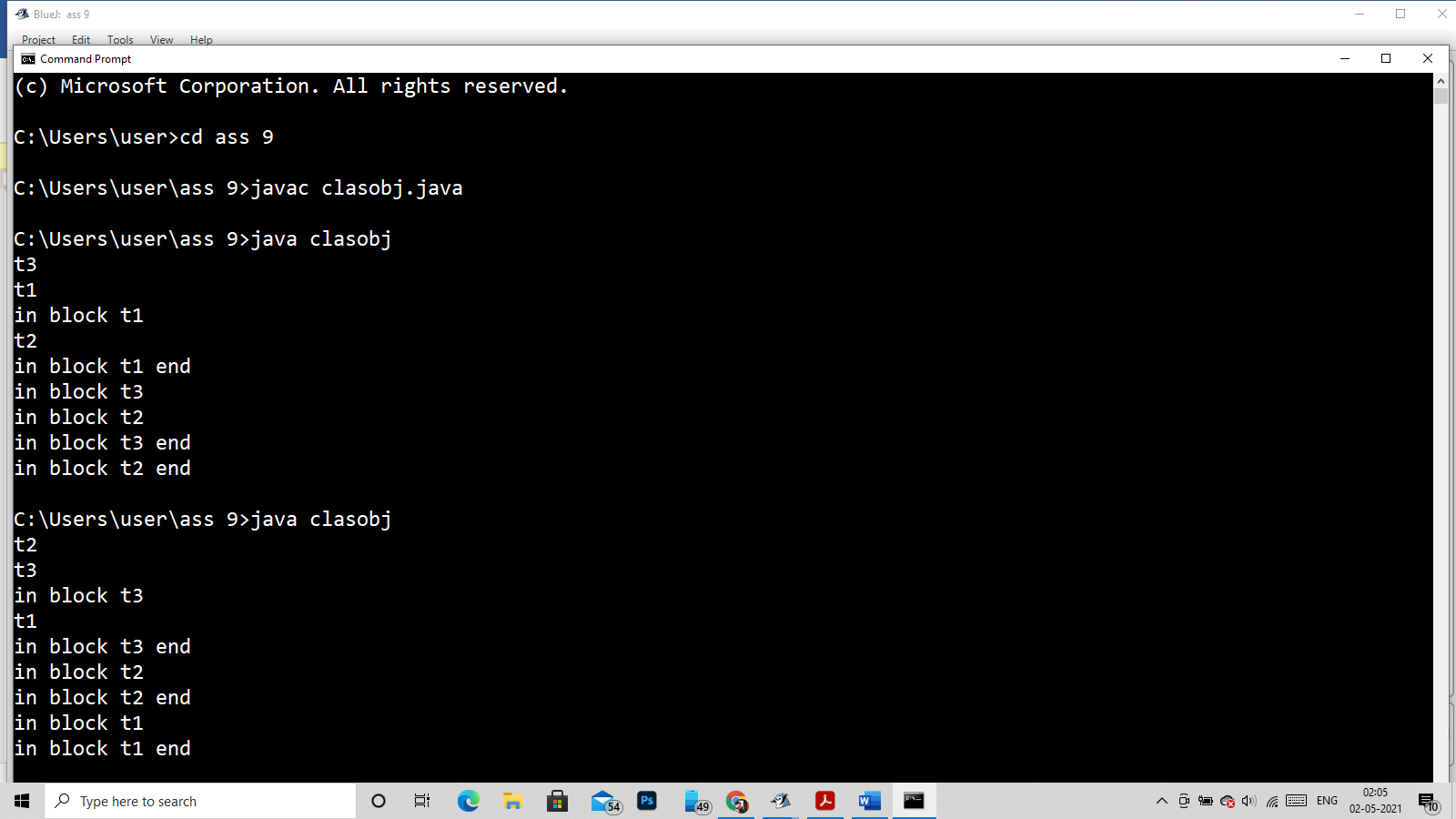
t1.start();

t2.start();

t3.start();

}

}



Q7)

With-out synchronized

class Test

{

char a;

Test(char a)

{

this.a=a;

}

public void printl(int k)

{

for(int i = 1;i<=5;i++)

{

System.out.println(a+"--->"+k\*i);

}

}

}

class A extends Thread

{

public void run()

{

new Test('A').printl(5);

}

}

class B extends Thread

{

public void run()

{

new Test('B').printl(100);

}

}

public class checkSync

{

public static void main(String asdf[])

{

A a = new A();

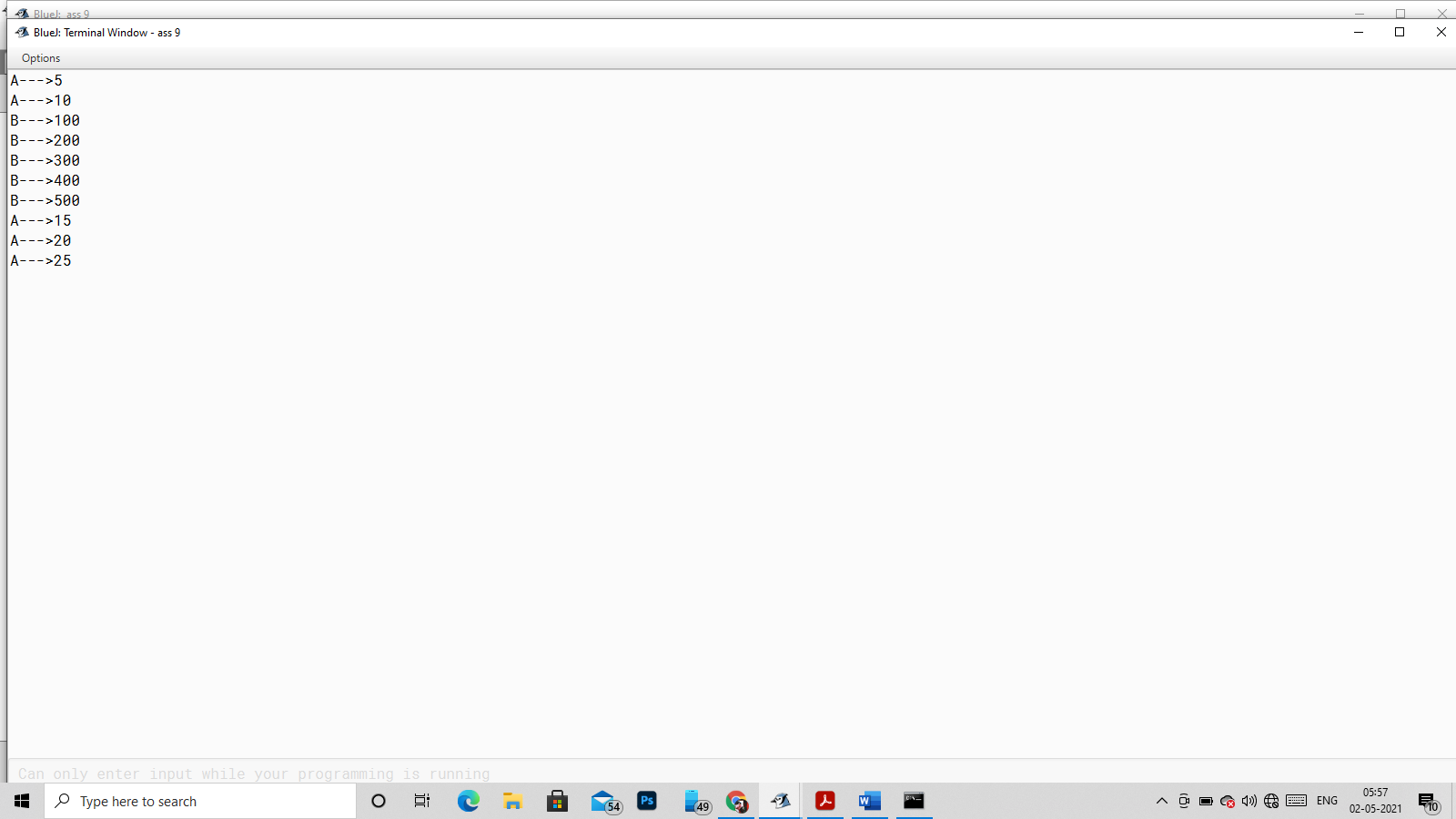
B b = new B();

a.start();

b.start();

}

}



With Synchronized

class Test

{

char a;

Test(char a)

{

this.a=a;

}

public synchronized void printl(int k)

{

for(int i = 1;i<=5;i++)

{

System.out.println(a+"--->"+k\*i);

}

}

}

class A extends Thread

{

public void run()

{

new Test('A').printl(5);

}

}

class B extends Thread

{

public void run()

{

new Test('B').printl(100);

}

}

public class checkSync

{

public static void main(String asdf[])

{

A a = new A();

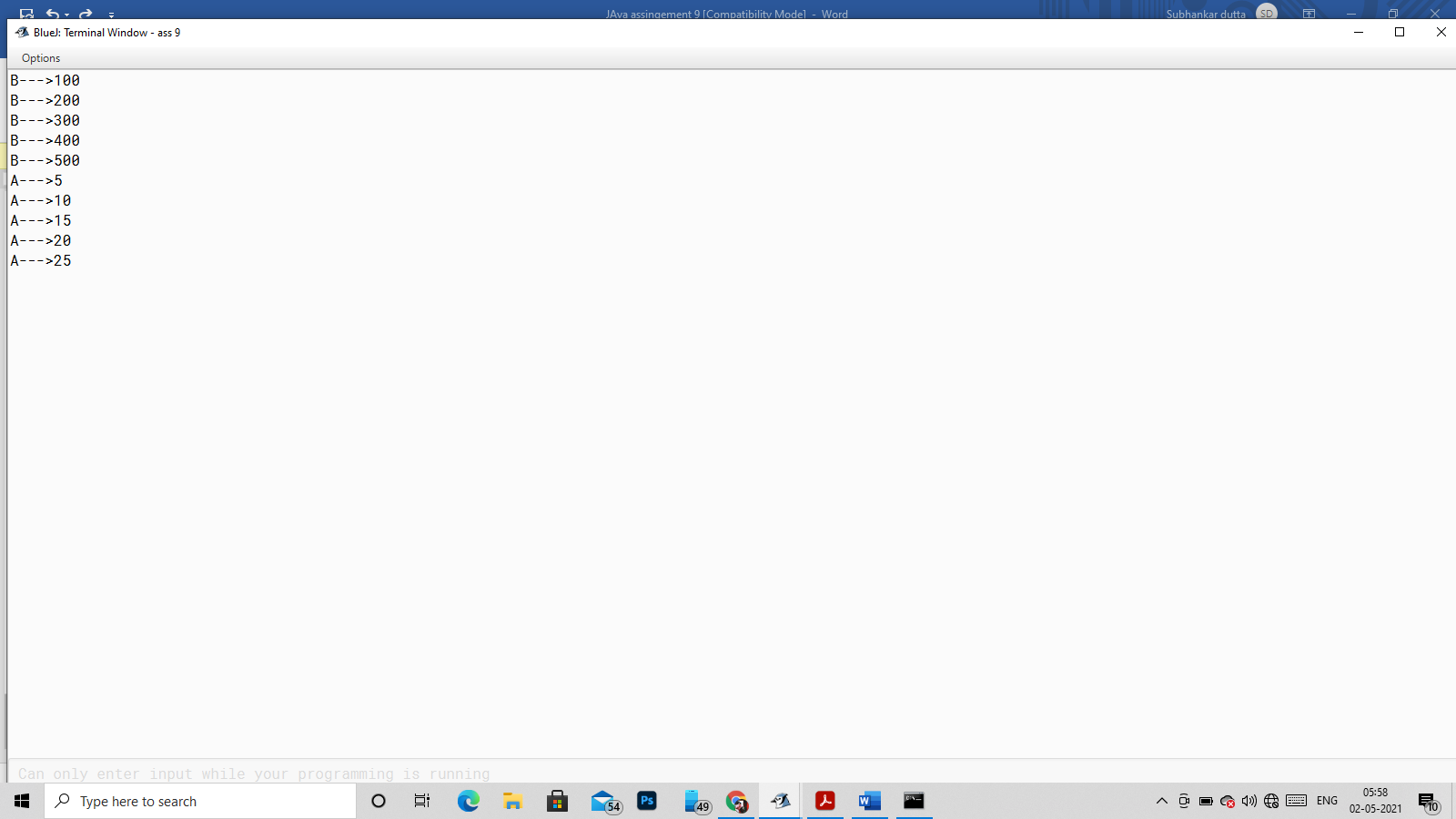
B b = new B();

a.start();

b.start();

}

}



Q9)

class Test

{

char a;

Test(char a)

{

this.a=a;

}

public void printl(int k)

{

synchronized(this)

{

for(int i = 1;i<=5;i++)

{

System.out.println(a+"--->"+k\*i);

try

{

Thread.sleep(1000);

}catch(Exception e)

{

System.out.println(e);

}

}

}

}

}

class A extends Thread

{

Test t;

A(Test t)

{

this.t= t;

}

public void run()

{

t.printl(5);

}

}

class B extends Thread

{

Test t;

B(Test t)

{

this.t= t;

}

public void run()

{

t.printl(100);

}

}

public class checkSync

{

public static void main(String asdf[])

{

Test t = new Test('A');

A a = new A(t);

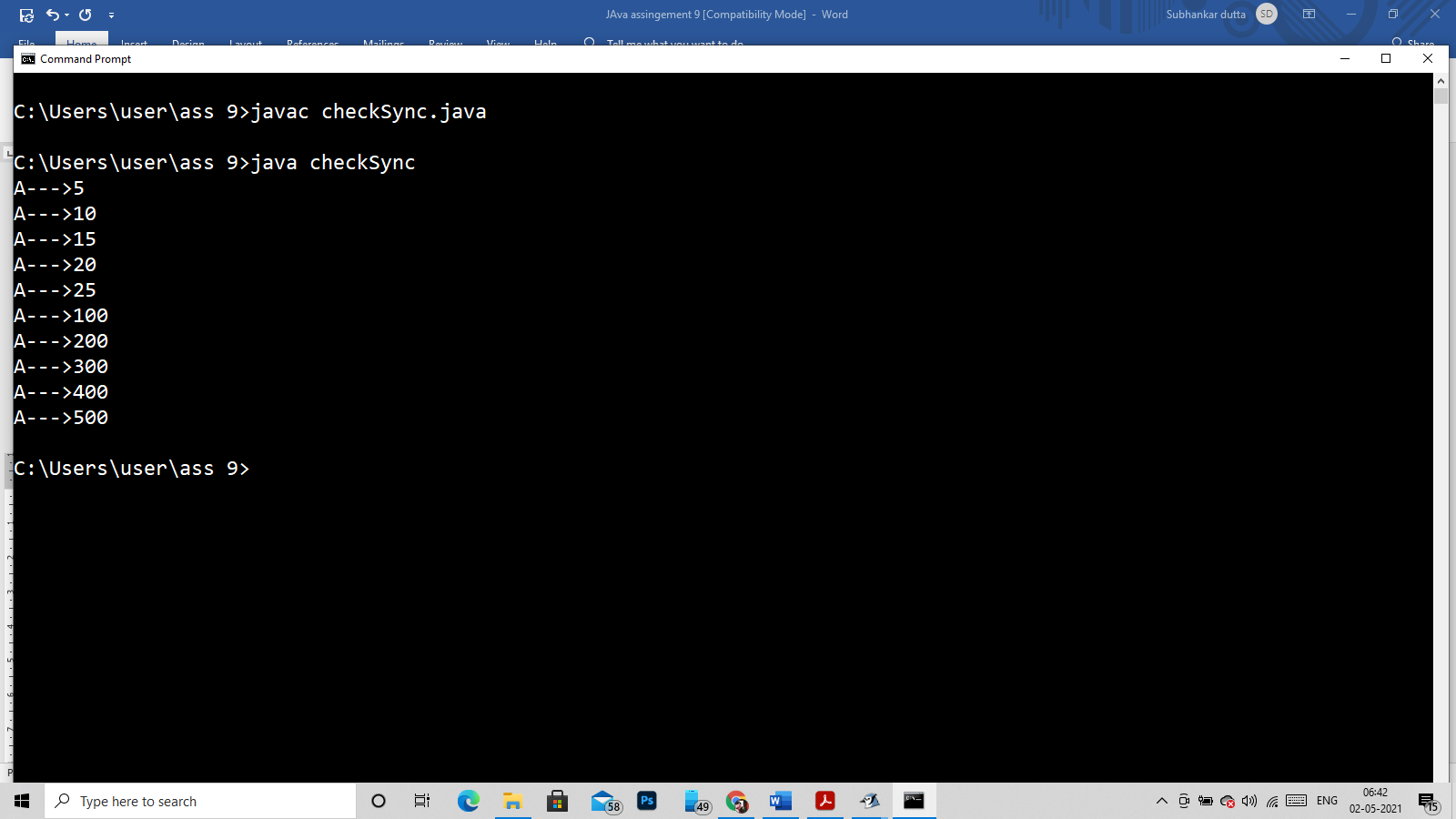
B b = new B(t);

a.start();

b.start();

}

}



Q10)

public class checkSync

{

public static void main(String asdf[])throws Exception

{

Object o1 = new Object();

Object o2 = new Object();

Object o3 = new Object();

Thread thread1 = new Thread(new deadLock(o1, o2), "thread1");

Thread thread2 = new Thread(new deadLock(o2, o3), "thread2");

thread1.start();

Thread.sleep(2000);//executing thread2

thread2.start();

Thread.sleep(2000);

}

}

class deadLock implements Runnable

{

Object obj1;

Object obj2;

deadLock(Object o1,Object o2)

{

obj1=o1;

obj2 = o2;

}

@Override

public void run()

{

String nm = Thread.currentThread().getName();

System.out.println(nm + "Acquire lock on "+obj1);

synchronized(obj1)

{

System.out.println(nm + "Acquired lock on "+obj1);

try

{

Thread.sleep(5000);

}catch(Exception e)

{

System.out.println("Error -->> "+e);

}

}

System.out.println(nm + "Relized lock on "+obj1);

System.out.println(nm + "Acquire lock on "+obj2);

synchronized(obj2)

{

System.out.println(nm + "Acquired lock on "+obj2);

try

{

Thread.sleep(5000);

}catch(Exception e)

{

System.out.println("Error -->> "+e);

}

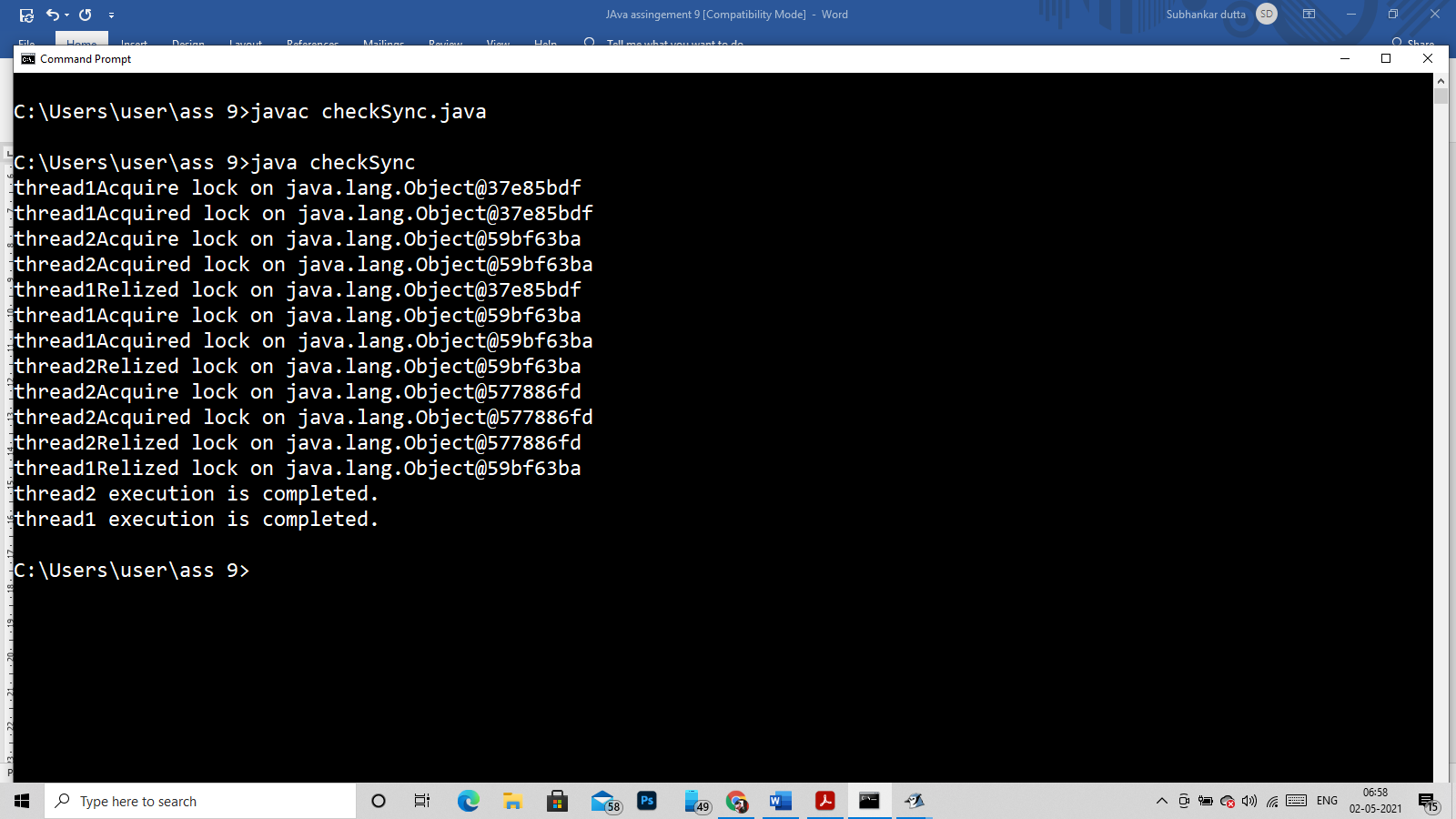
}

System.out.println(nm + "Relized lock on "+obj2);

System.out.println(nm + " execution is completed.");

}

}



Q11)

public class checkSync {

public static void main(String[] args) {

final String resource1 = "Binod Kumar";

final String resource2 = "Bipin Kumar";

Thread t1 = new Thread() {

public void run() {

synchronized (resource1) {

System.out.println("Thread 1: locked resource 1");

try { Thread.sleep(100);} catch (Exception e) {}

synchronized (resource2) {

System.out.println("Thread 1: locked resource 2");

}

}

}

};

Thread t2 = new Thread() {

public void run() {

synchronized (resource2) {

System.out.println("Thread 2: locked resource 2");

try { Thread.sleep(100);} catch (Exception e) {}

synchronized (resource1) {

System.out.println("Thread 2: locked resource 1");

}

}

}

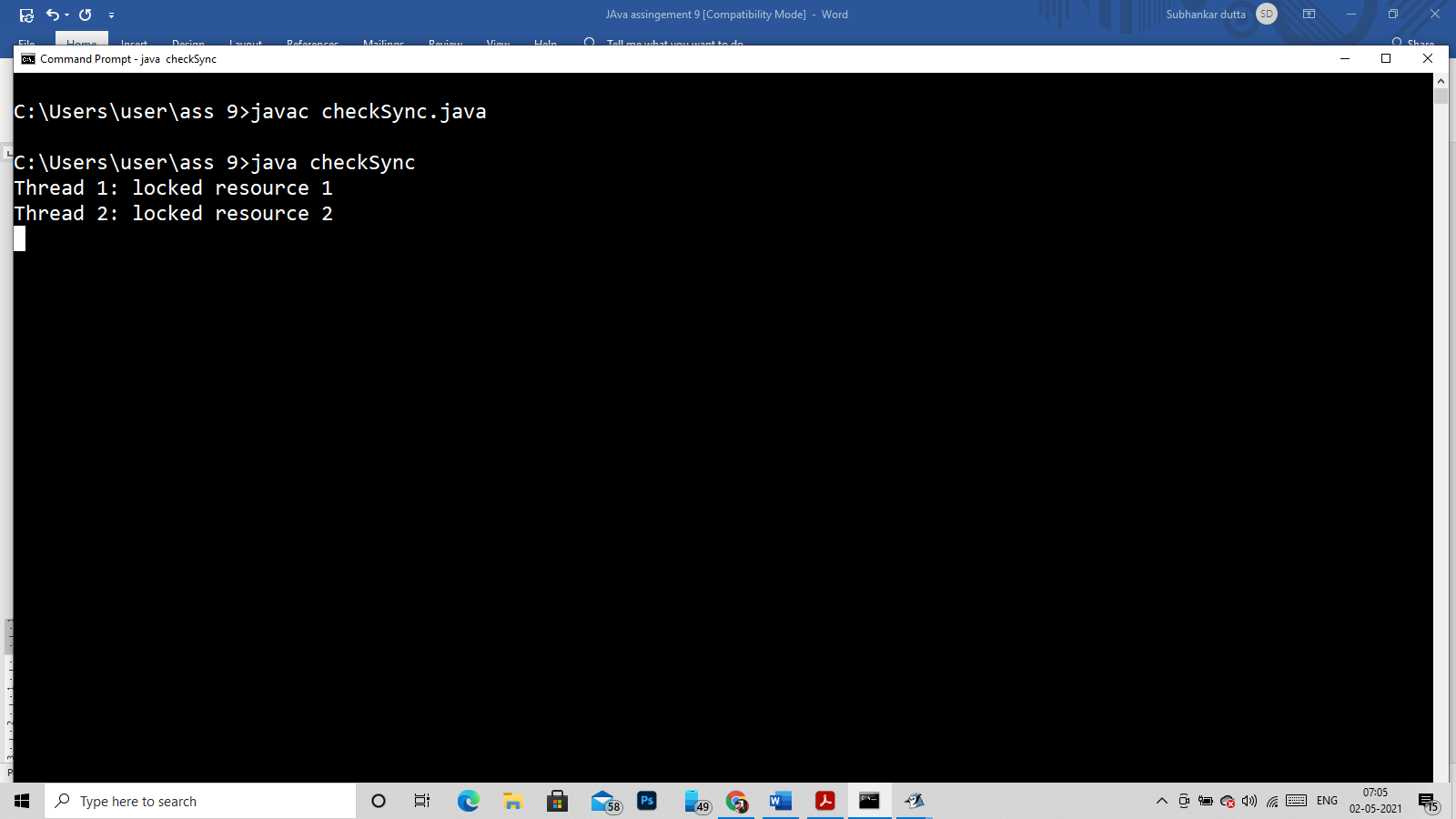
};

t1.start();

t2.start();

}

}



JVM got stuck after this execution

Way to clear deadlock using wait and notify…!!!

public class checkSync {

public static void main(String[] args) {

Cust c = new Cust();

new Thread(){

public void run()

{

c.withdraw(100);}

}.start();

new Thread(){

public void run()

{

c.deposit(1000);}

}.start();

}

}

class Cust

{

int amt = 1000;

synchronized void withdraw(int a)

{

if(a<amt)

{

System.out.println("Less Balance... Wait for deposit");

try

{

wait();

Thread.sleep(1000);

}catch(Exception e)

{

System.out.println(e);

}

}

a=a-amt;

System.out.println("Withdraw copleated!!!!");

}

synchronized void deposit(int a)

{

System.out.println("Going to deposit....");

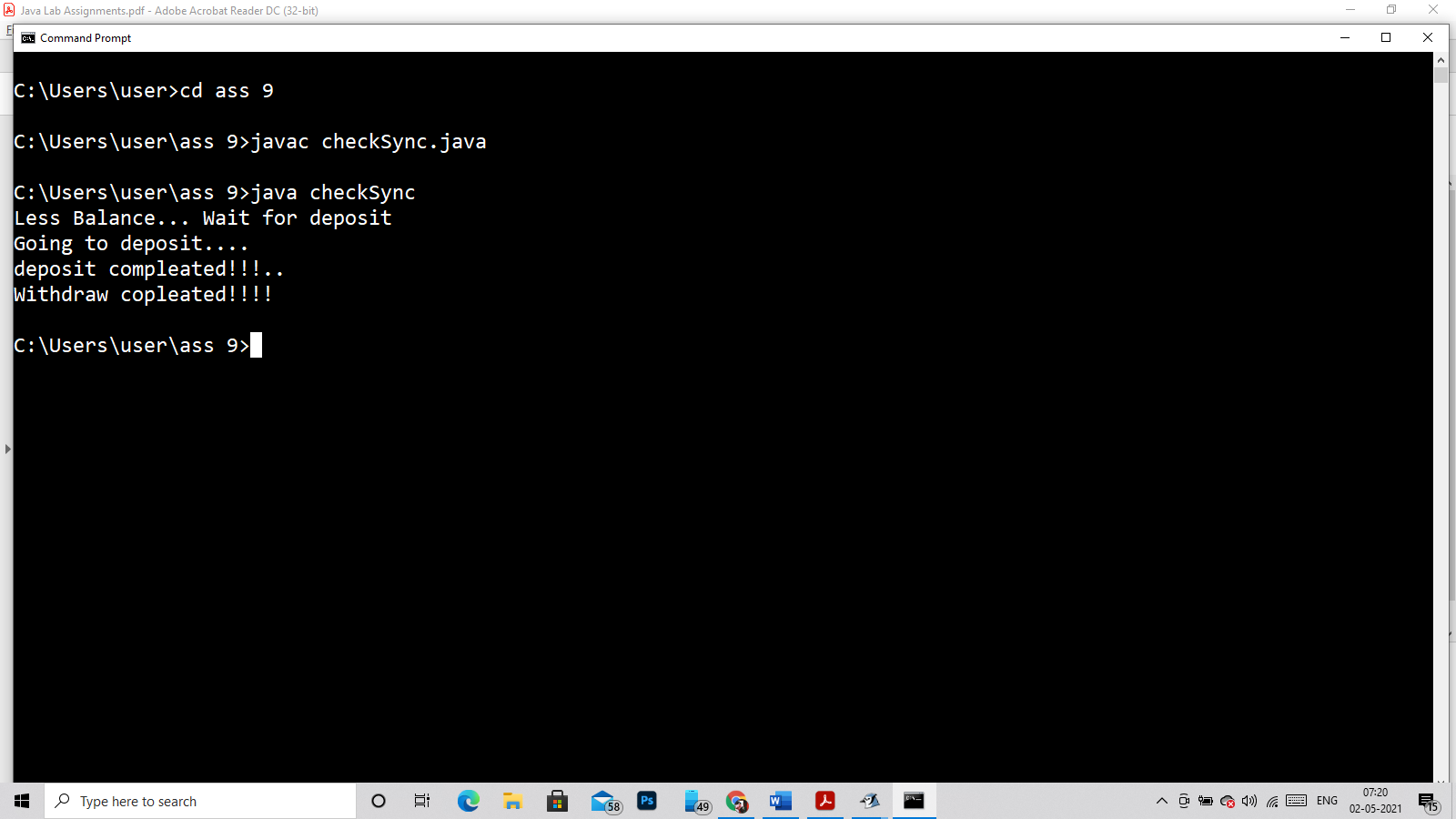
amt=amt+a;

System.out.println("deposit compleated!!!..");

notify();

}

}



Q12)

class A implements Runnable

{

public void run()

{

for(int i = 1;i<=5;i++)

{

System.out.println(i);

try

{

Thread.sleep(1000);

}

catch(Exception e)

{

System.out.println(e);

}

}

}

}

public class checkSync

{

public static void main(String args[])

{

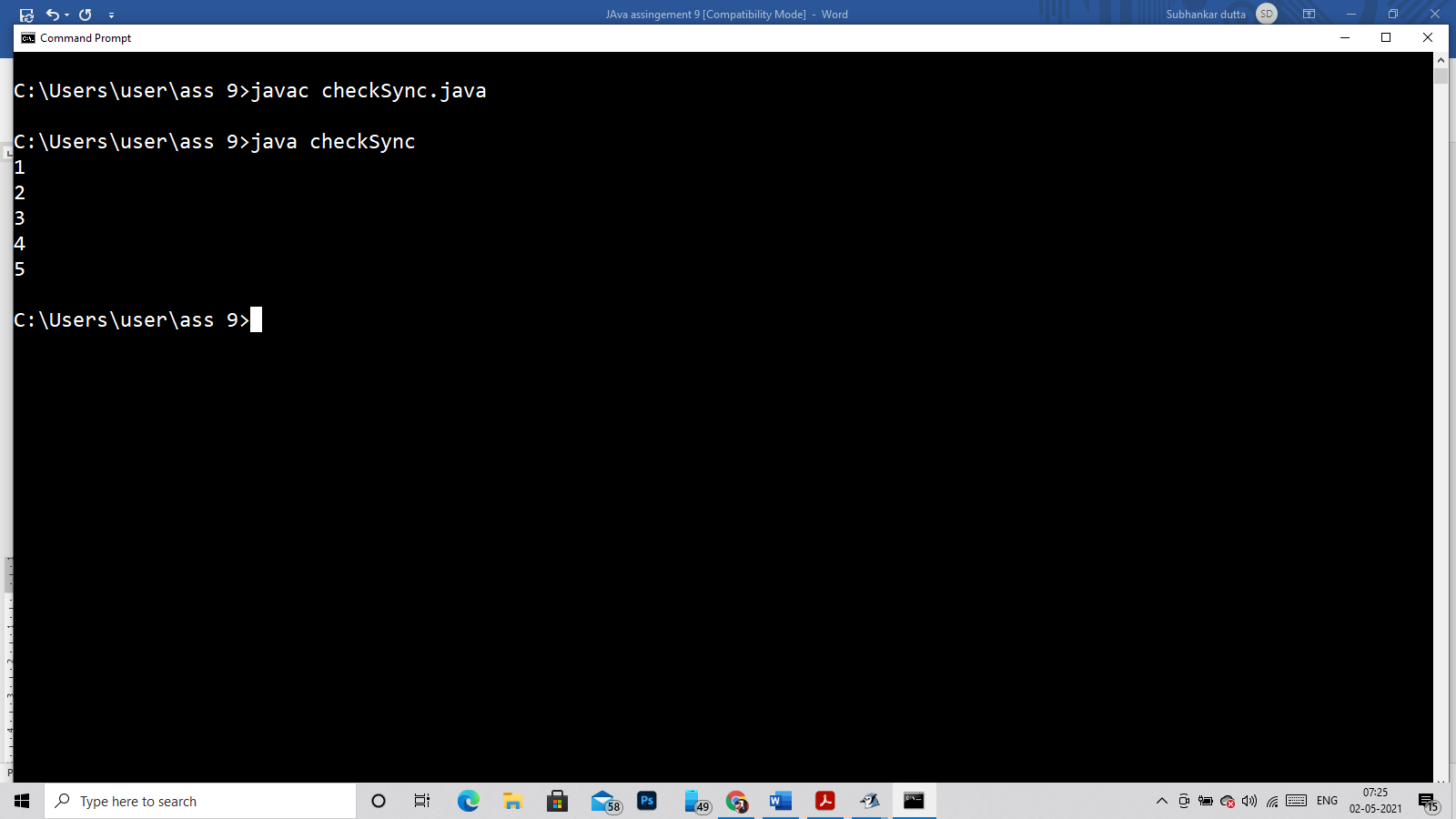
A a = new A();

Thread t1 = new Thread(a);

t1.start();

}

}



Q13, Q15, Q17)

class A implements Runnable

{

public void run()

{

for(int i = 1;i<=5;i++)

{

System.out.println(Thread.currentThread().getName()+"--->>"+i);

try

{

Thread.sleep(1000);

}

catch(Exception e)

{

System.out.println(e);

}

}

}

}

public class checkSync

{

public static void main(String args[])

{

A a = new A();

Thread t1 = new Thread(a);

Thread t2 = new Thread(a);

Thread t3 = new Thread(a);

Thread t4 = new Thread(a);

t1.setPriority(1); //Question number 15

t2.setPriority(3); //Question number 15

t3.setPriority(5); //Question number 15

t4.setPriority(8); //Question number 15

System.out.println(t1.getName()+ " Priority is -->>> "+t1.getPriority());

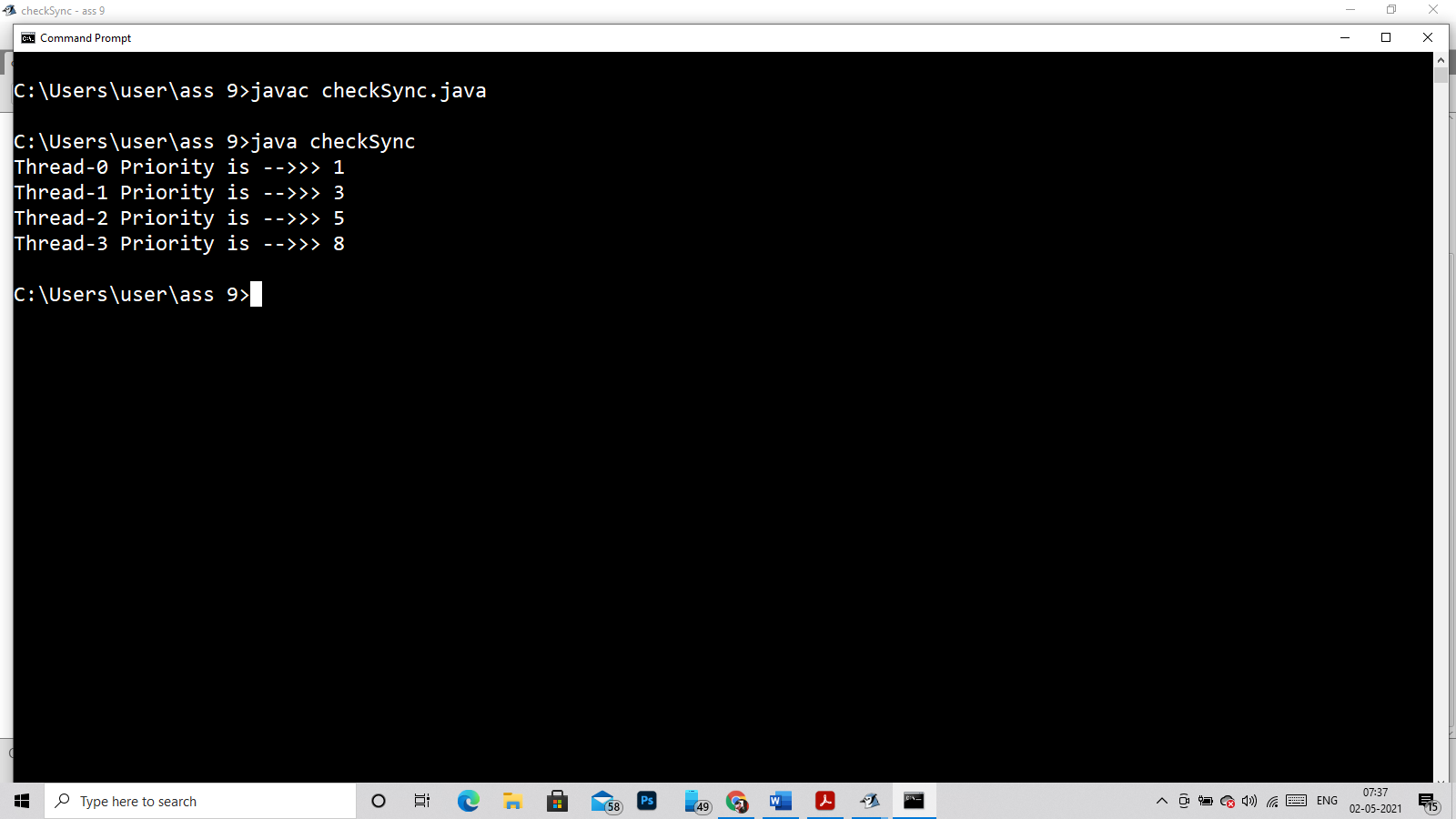
System.out.println(t2.getName()+ " Priority is -->>> "+t2.getPriority());

System.out.println(t3.getName()+ " Priority is -->>> "+t3.getPriority());

System.out.println(t4.getName()+ " Priority is -->>> "+t4.getPriority());

}

}



Q14)

class A implements Runnable

{

public void run()

{

for(int i = 1;i<=5;i++)

{

System.out.println(Thread.currentThread().getName()+"--->>"+i);

try

{

Thread.sleep(4000);

}

catch(Exception e)

{

System.out.println(e);

}

}

}

}

public class checkSync

{

public static void main(String args[])

{

A a = new A();

Thread t1 = new Thread(a);

Thread t2 = new Thread(a);

Thread t3 = new Thread(a);

t1.setPriority(Thread.MIN\_PRIORITY);

t2.setPriority(Thread.NORM\_PRIORITY);

t3.setPriority(Thread.MAX\_PRIORITY);

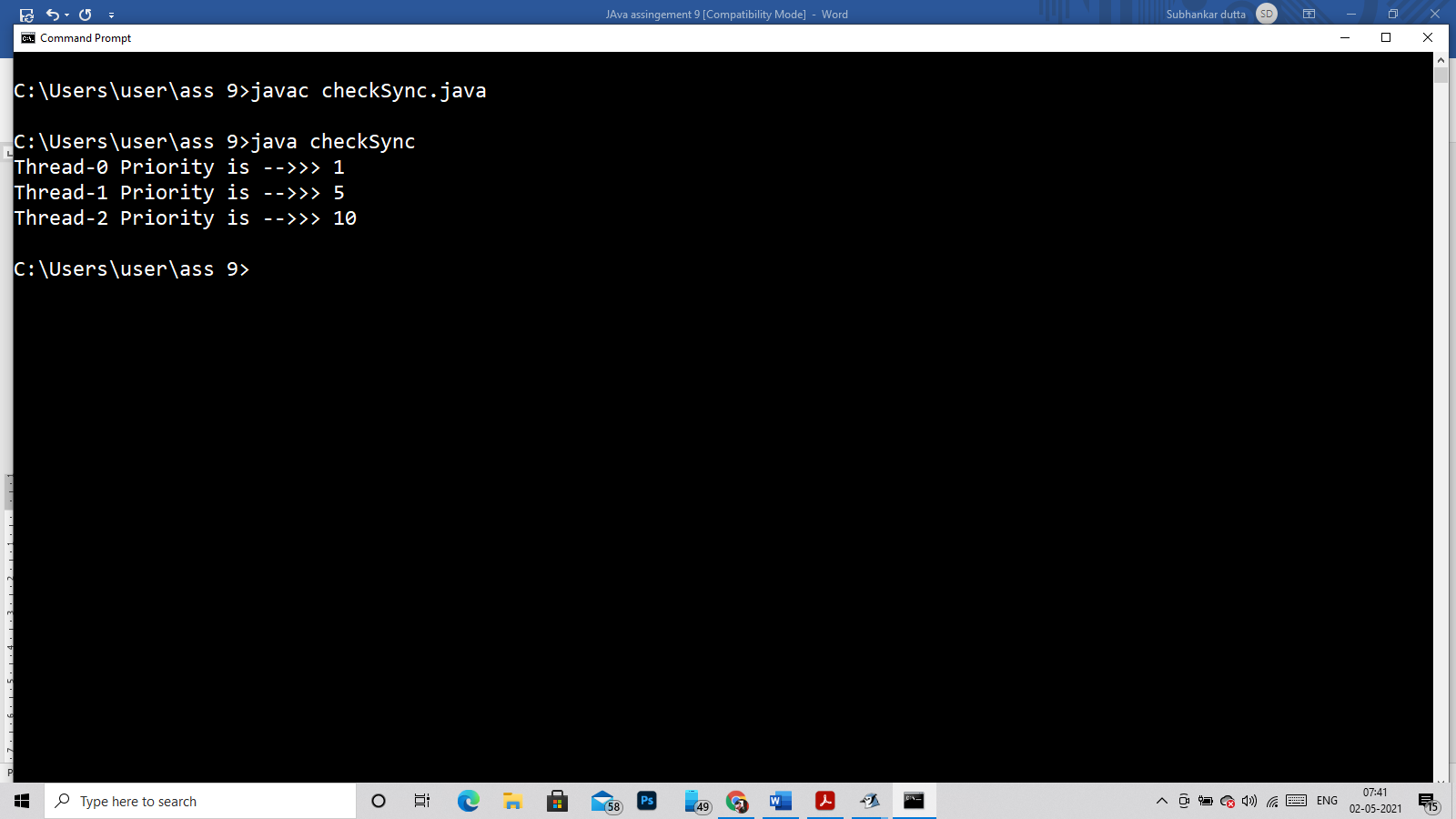
System.out.println(t1.getName()+ " Priority is -->>> "+t1.getPriority());

System.out.println(t2.getName()+ " Priority is -->>> "+t2.getPriority());

System.out.println(t3.getName()+ " Priority is -->>> "+t3.getPriority());

}

}



Q16)

class A implements Runnable

{

public void run()

{

for(int i = 1;i<=5;i++)

{

System.out.println(Thread.currentThread().getName()+"--->>"+i+" priority -->> "+Thread.currentThread().getPriority());

try

{

Thread.sleep(4000);

}

catch(Exception e)

{

System.out.println(e);

}

}

}

}

public class checkSync

{

public static void main(String args[])

{

A a = new A();

Thread t1 = new Thread(a);

Thread t2 = new Thread(a);

Thread t3 = new Thread(a);

t1.setPriority(Thread.MIN\_PRIORITY);

t2.setPriority(Thread.NORM\_PRIORITY);

t3.setPriority(Thread.MAX\_PRIORITY);

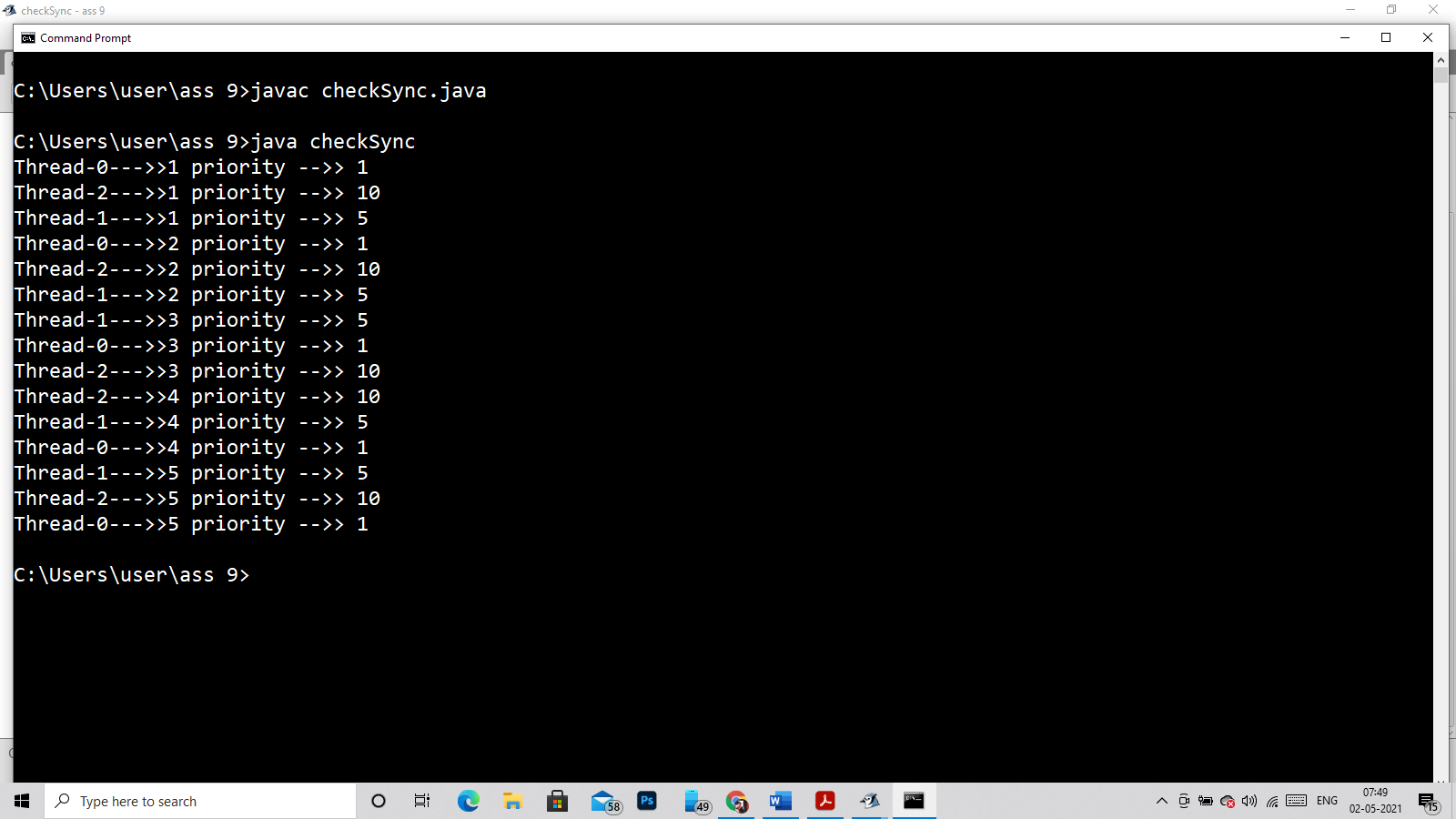
t1.start();

t2.start();

t3.start();

}

}



Q18)

class A implements Runnable

{

public void run()

{

for(int i = 1;i<=5;i++)

{

System.out.println(Thread.currentThread().getName()+"--->>"+i);

try

{

Thread.sleep(400);

}

catch(Exception e)

{

System.out.println(e);

}

}

}

}

public class checkSync

{

public static void main(String args[])

{

A a = new A();

Thread t1 = new Thread(a);

Thread t2 = new Thread(a);

Thread t3 = new Thread(a);

t1.start();

try

{

t1.join(2500);

}catch(Exception e)

{

}

t2.start();

try

{

t2.join(2500);

}catch(Exception e)

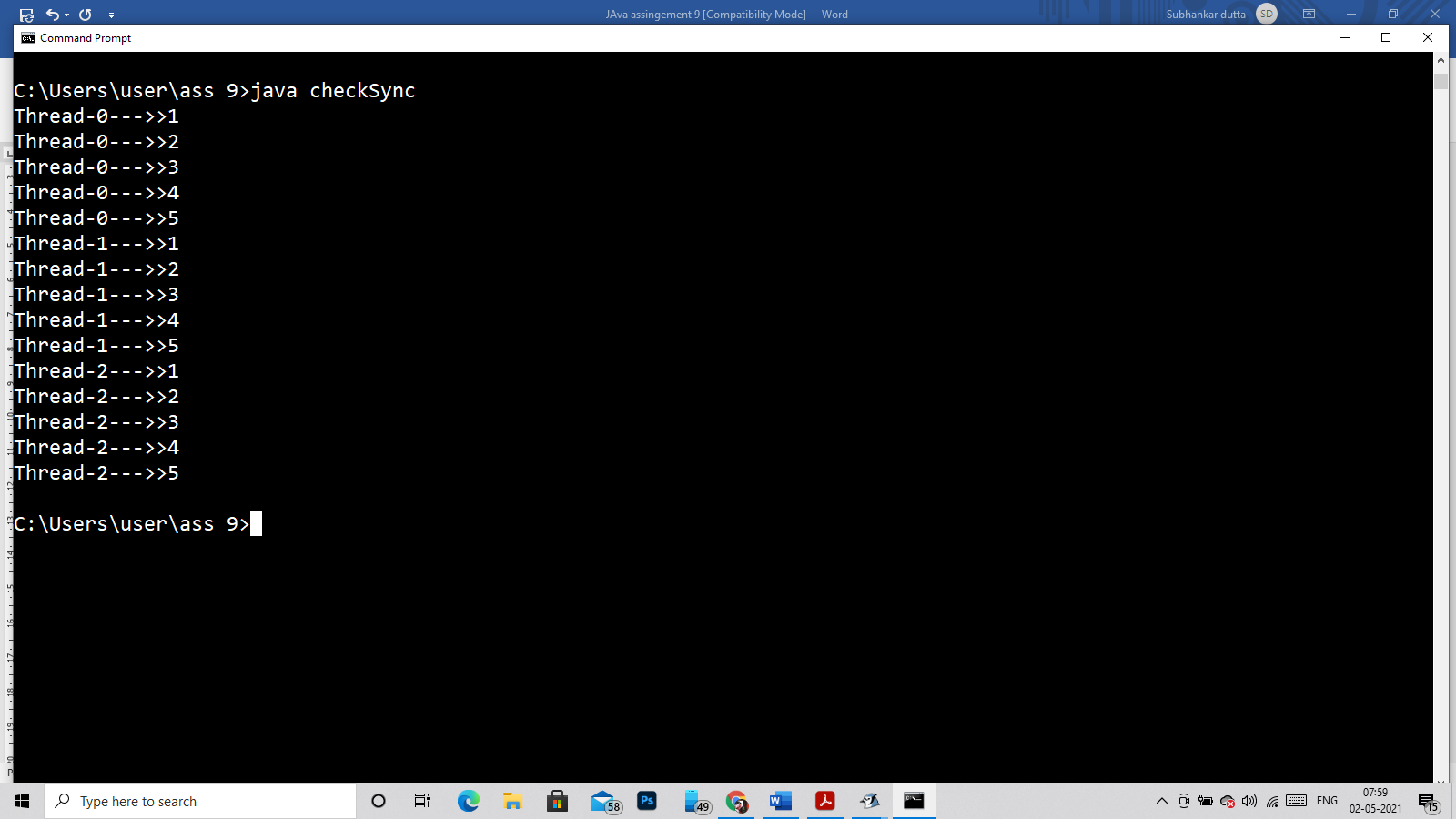
{

}

t3.start();

}

}



Q19)

class Check extends Thread

{

public void run()

{

System.out.println("I am under Run Thread!!!!");

}

}

public class Ext

{

public static void main(String args[])

{

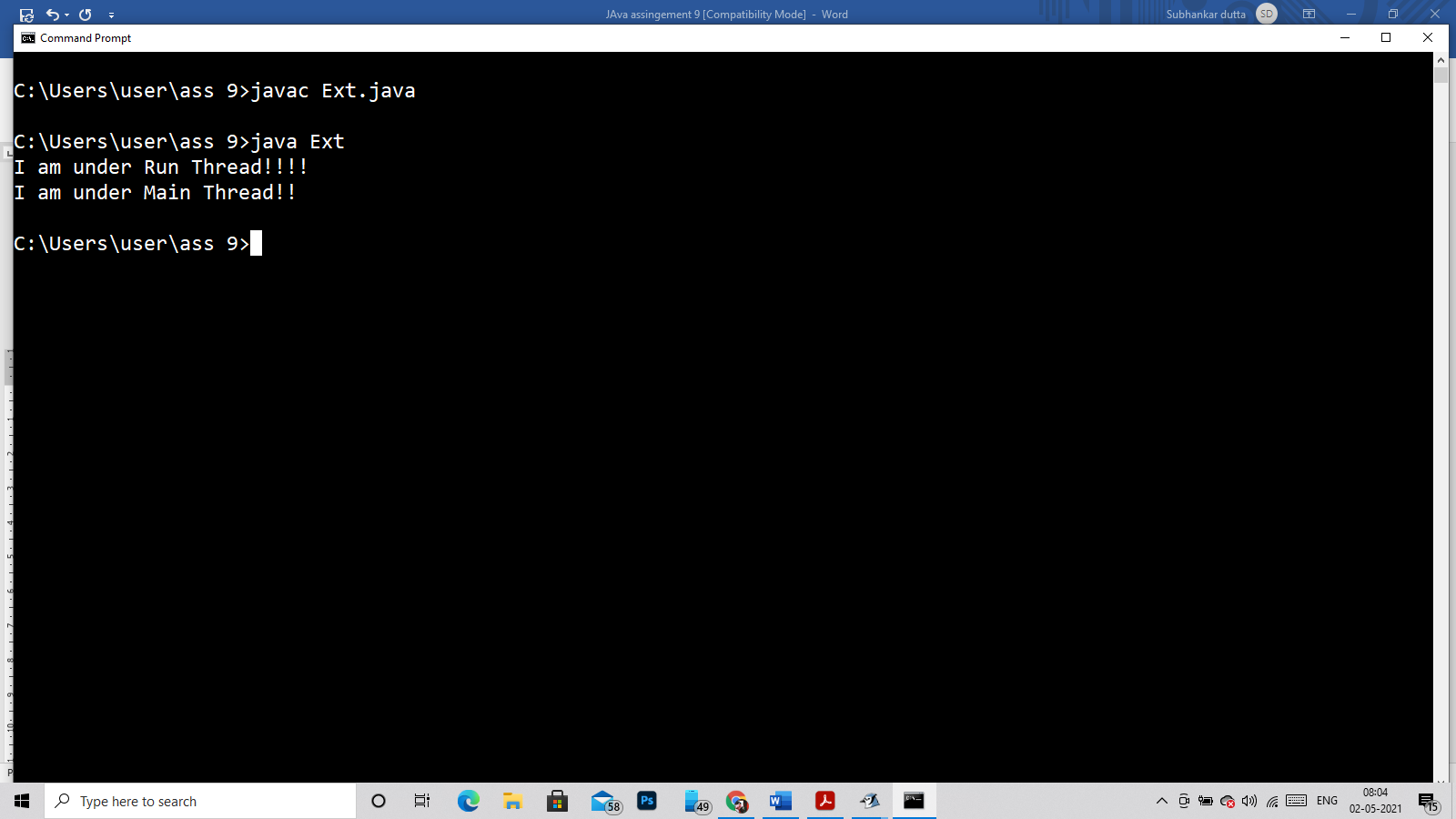
Check o = new Check();

o.run();

System.out.println("I am under Main Thread!!");

}

}



Q20)

Error:::

public class Demo extends Thread

{

public static void main(String args[])

{

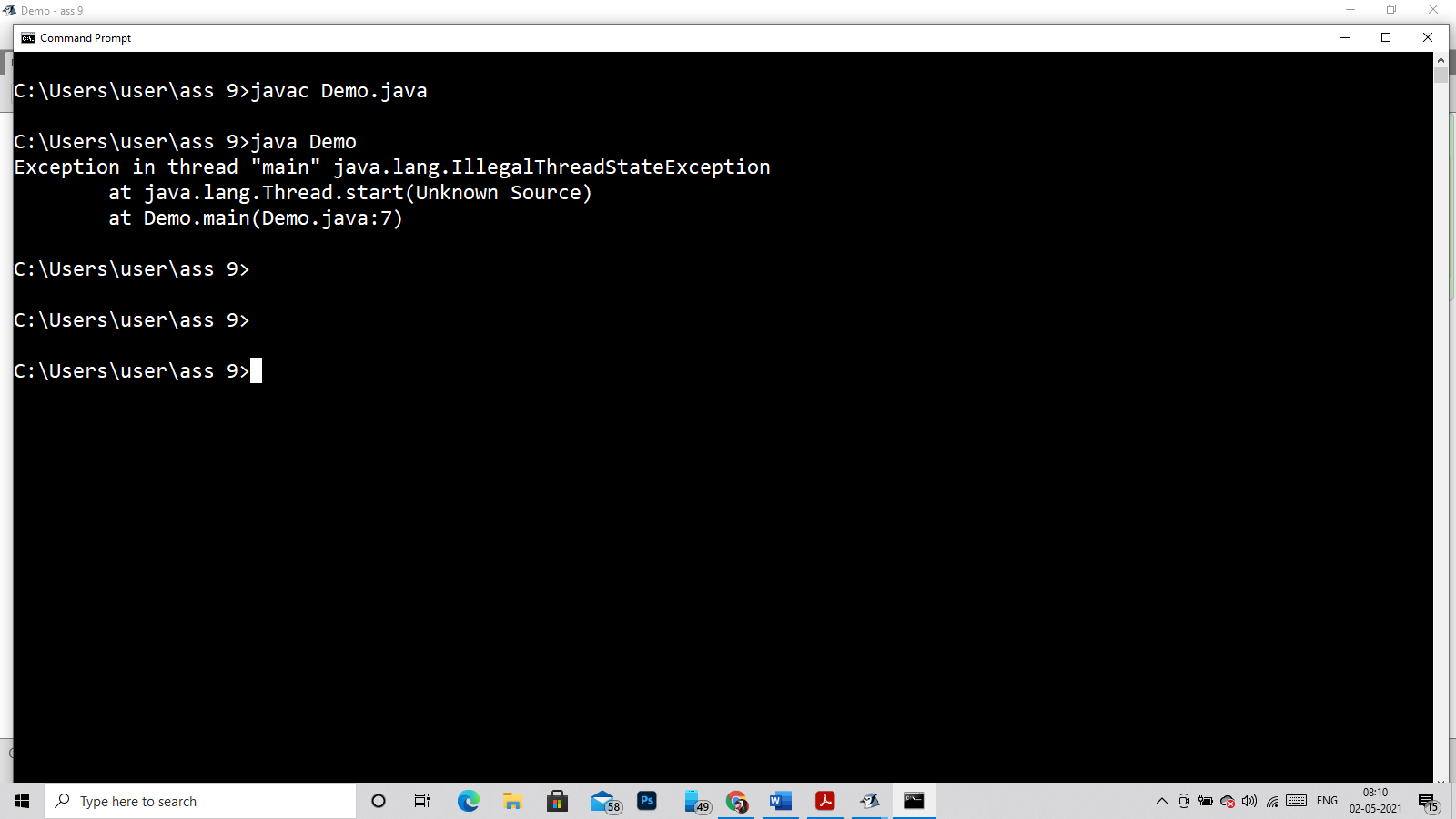
Demo d1 = new Demo();

d1.start();

d1.start();

}

}



After handeling it

public class Demo extends Thread

{

public static void main(String args[])

{

try

{

Demo d1 = new Demo();

d1.start();

d1.start();

}catch(IllegalThreadStateException e)

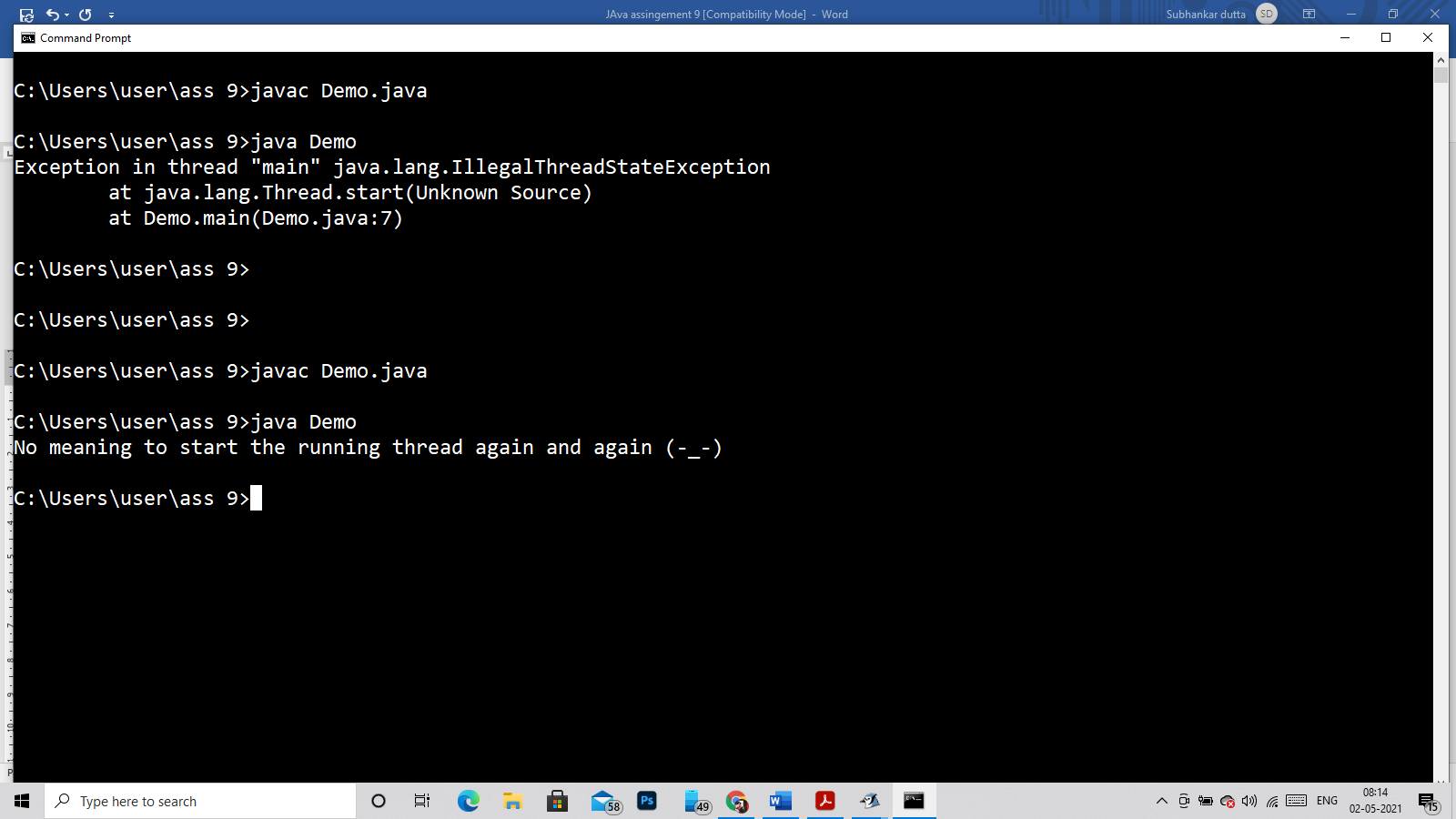
{

System.out.println("No meaning to start the running thread again and again (-\_-)");

}

}

}



Q21

class Student{

int rollno;

String name;

static String college;

Student(int r, String n,String c){

rollno = r;

name = n;

college = c;

}

void display (){System.out.println(rollno+" "+name+" "+college);}

}

public class StaticMain{

public static void main(String args[]){

Student s1 = new Student(1012,"Binod","IEM");

Student s2 = new Student(1220,"Bipin","UEM");

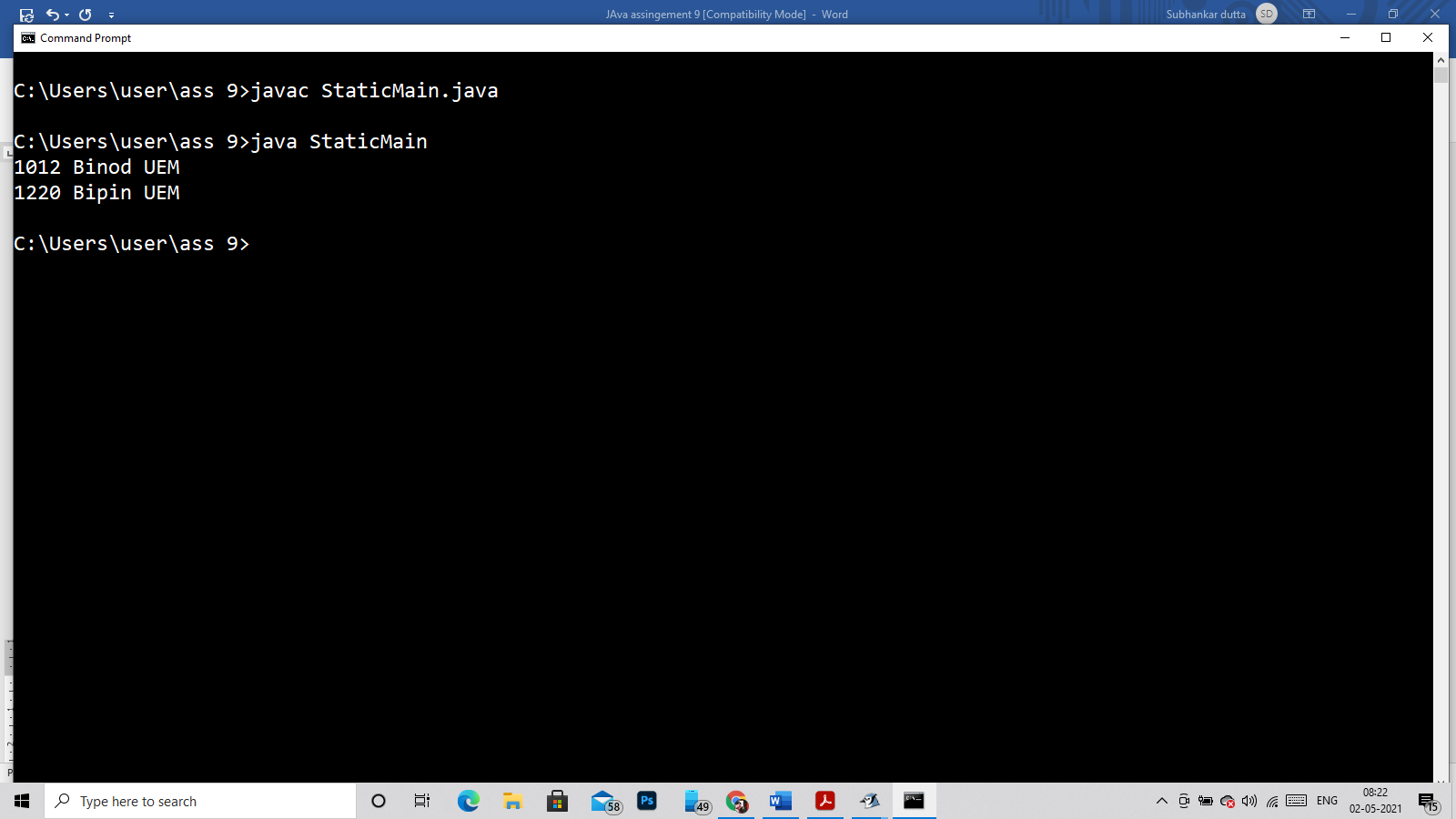
s1.display();

s2.display();

}

}

To. Be noted Binod read in IEM and Bipin read in UEM but after execution the both read in UEM due to static Var



Q22)

public class ProCk{

public static void main(String[] args)

{

int arr[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

for (int i = 0; i < arr.length; i++) {

if (arr[i] >= 4)

{

System.out.println("Exit from the loop and exceting JVM");

System.exit(0);

}

else

System.out.println("arr["+i+"] = " +arr[i]);

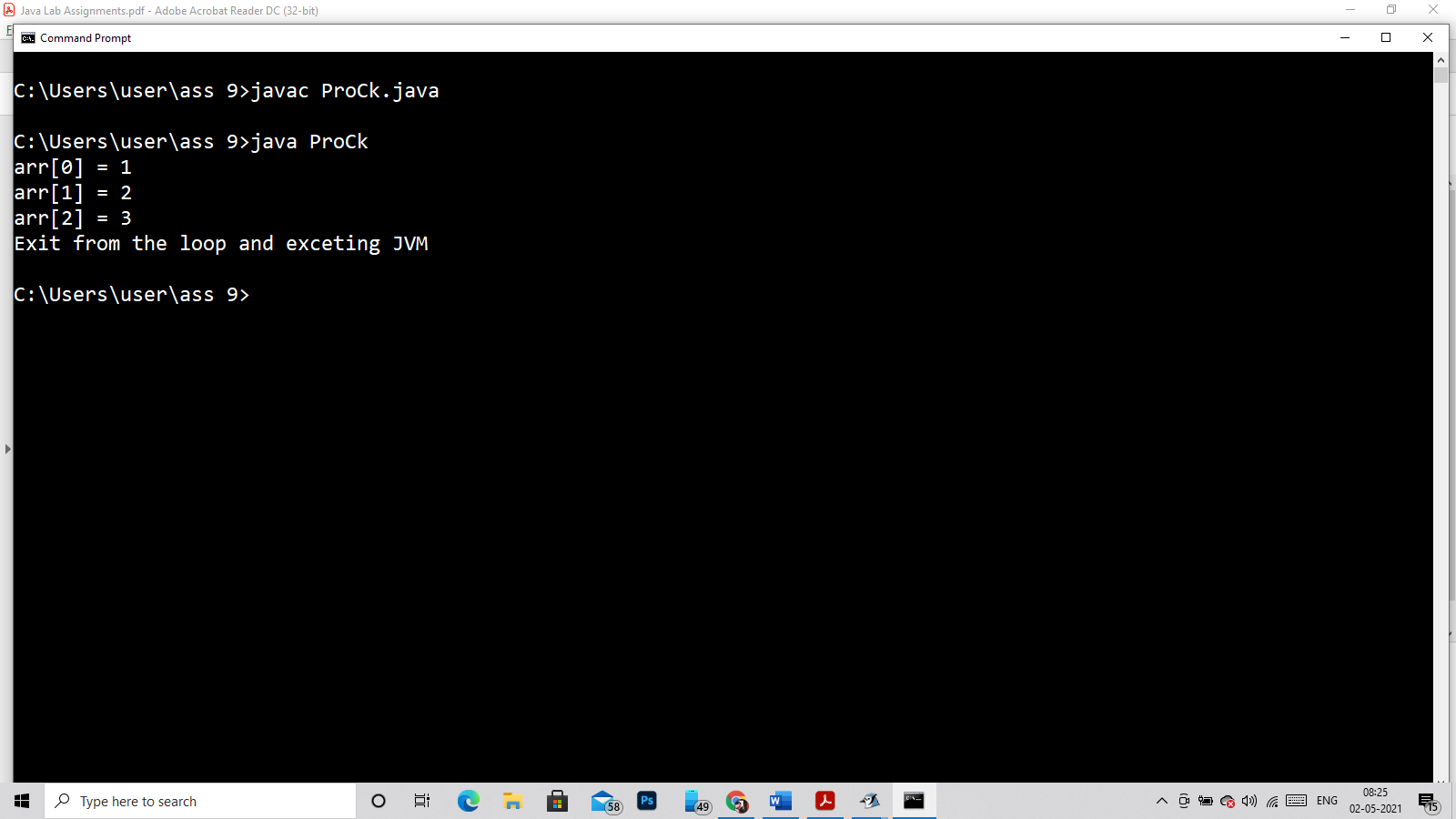
}

System.out.println("End of the Program");

}

}

Thus you will not see the Output “**End of the program**”.



Q23)

public class Demo extends Thread

{

public void run()

{

System.out.println(Thread.currentThread().getName());

}

public static void main(String args[])

{

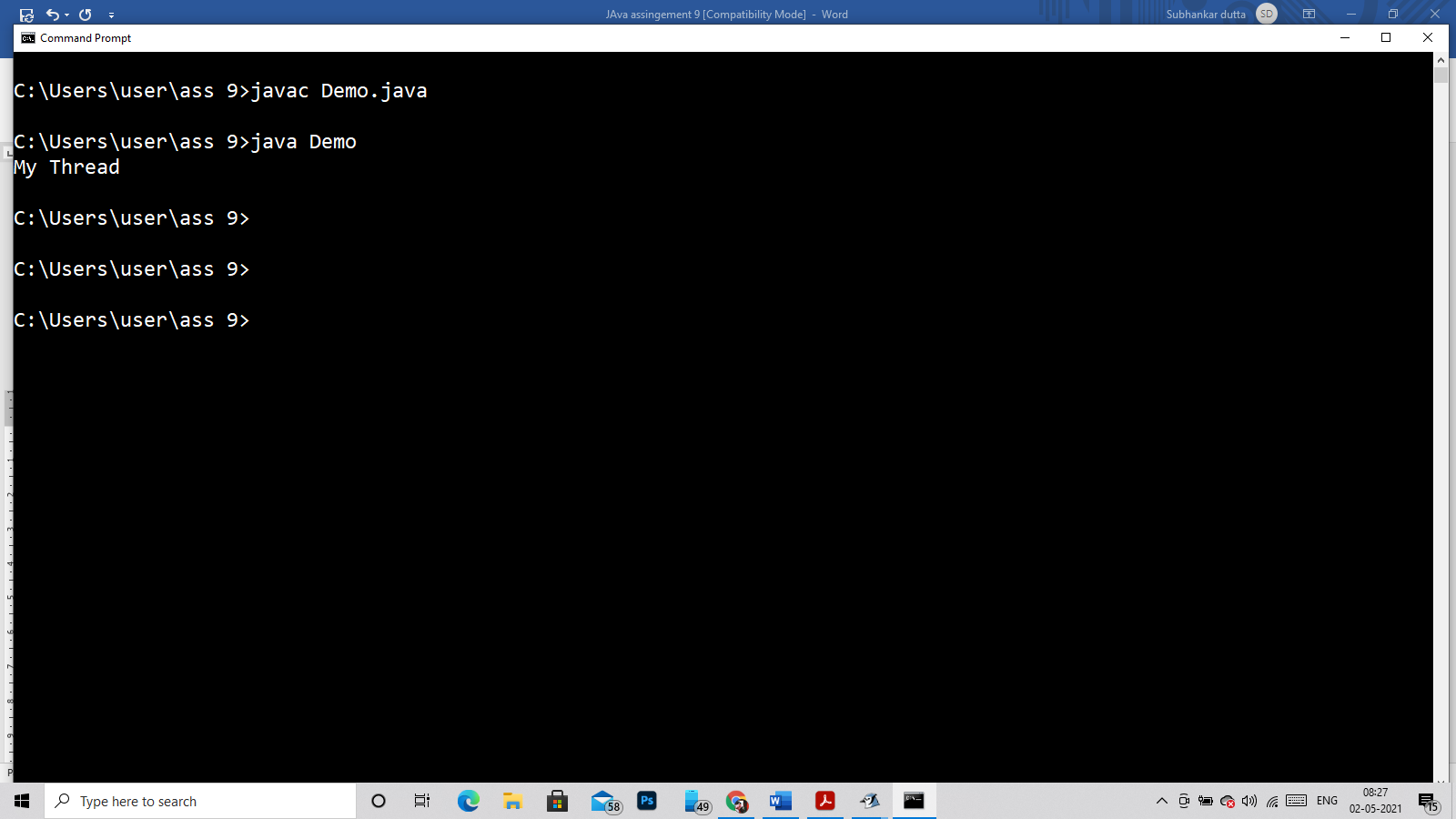
Demo d1 = new Demo();

d1.setName("My Thread");

d1.start();

}

}



Q24)

public class Destroy extends Thread

{

Destroy(String threadname, ThreadGroup tg)

{

super(tg, threadname);

start();

}

public void run()

{

for (int i = 0; i < 2; i++)

{

System.out.println(i);

try

{

Thread.sleep(1000);

}

catch (Exception ex) {

System.out.println("Exception encounterted");

}

}

System.out.println(Thread.currentThread().getName() + " finished executing");

}

public static void main(String arg[]) throws InterruptedException, SecurityException

{

ThreadGroup g1 = new ThreadGroup("Parent thread");

ThreadGroup g2 = new ThreadGroup(g1, "child thread");

Destroy t1 = new Destroy("Thread-1", g1);

Destroy t2 = new Destroy("Thread-2", g1);

t1.join();

t2.join();

g2.destroy();

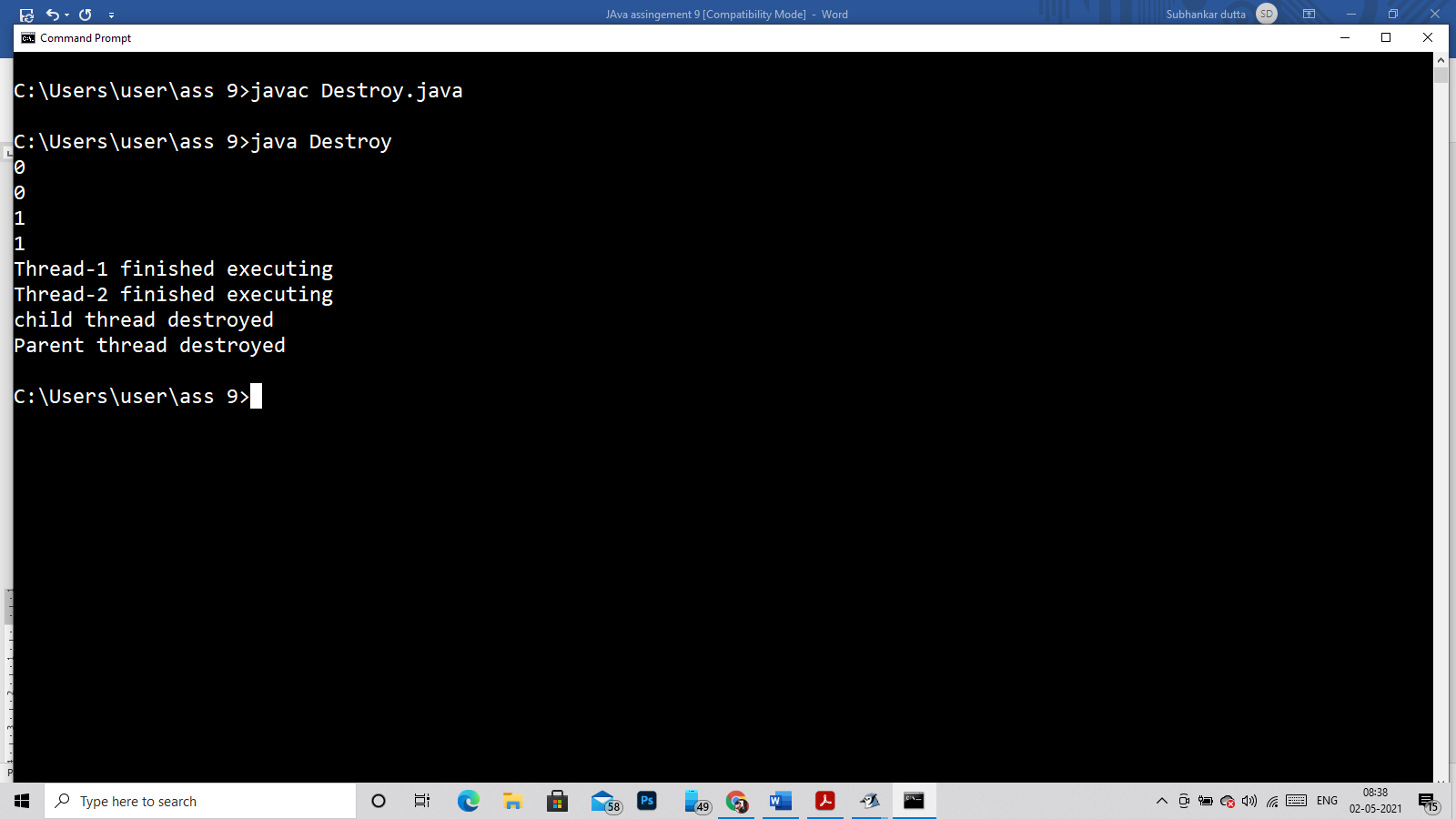
System.out.println(g2.getName() + " destroyed");

g1.destroy();

System.out.println(g1.getName() + " destroyed");

}

}



public class Suspend extends Thread

{

public void run()

{

for(int i=1; i<5; i++)

{

try

{

sleep(500);

System.out.println(Thread.currentThread().getName());

}catch(Exception e){System.out.println(e);}

System.out.println(i);

}

}

public static void main(String args[])

{

Suspend t1=new Suspend ();

Suspend t2=new Suspend ();

Suspend t3=new Suspend ();

t1.start();

t2.start();

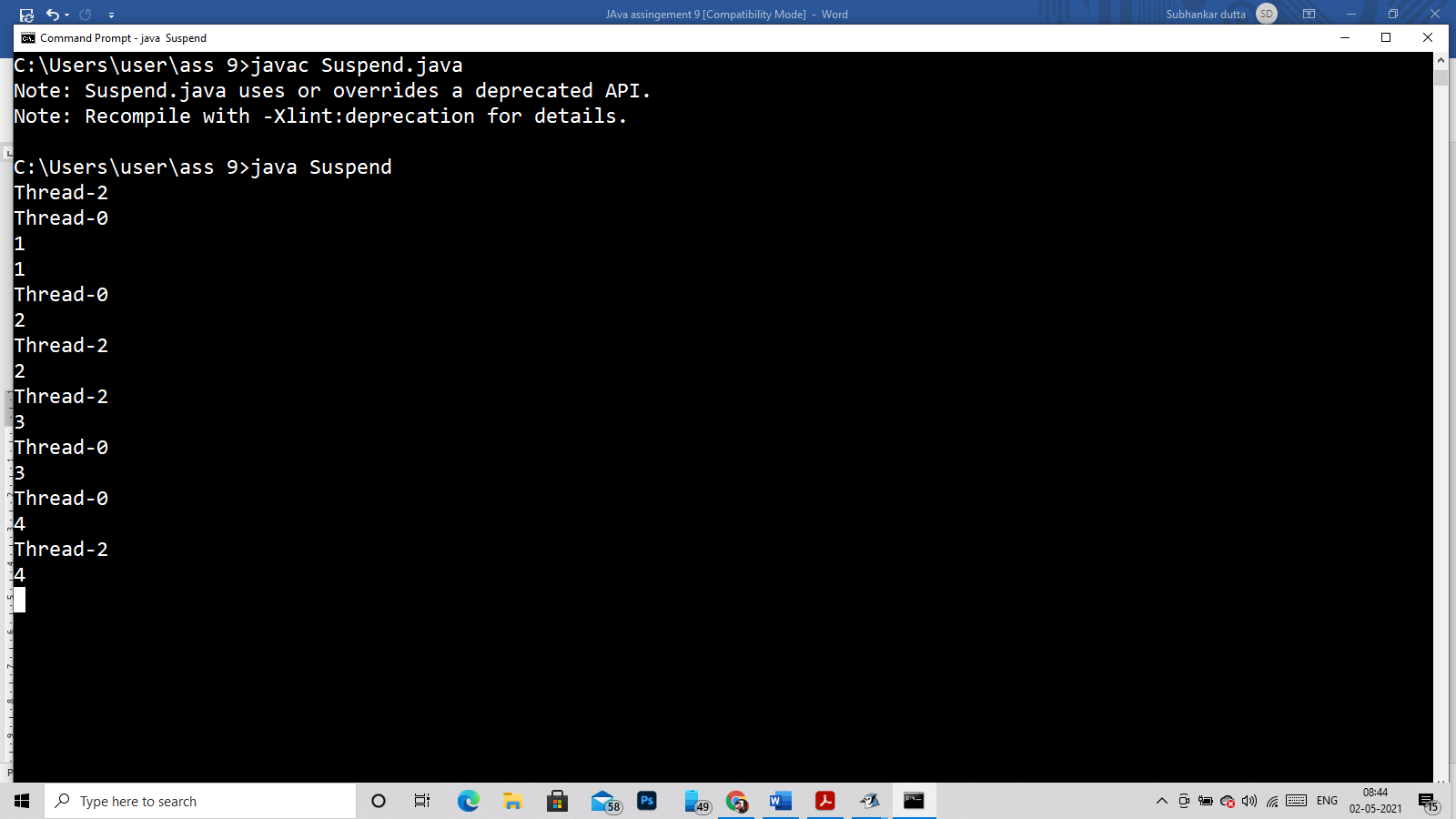
t2.suspend();

t3.start();

}

}

JVM Suspended!!!



Q26)

public class Suspend extends Thread

{

public void run()

{

for(int i=1; i<5; i++)

{

try

{

sleep(500);

System.out.println(Thread.currentThread());

}catch(Exception e){System.out.println(e);}

System.out.println(i);

}

}

public static void main(String args[])

{

Suspend t1=new Suspend ();

Suspend t2=new Suspend ();

Suspend t3=new Suspend ();

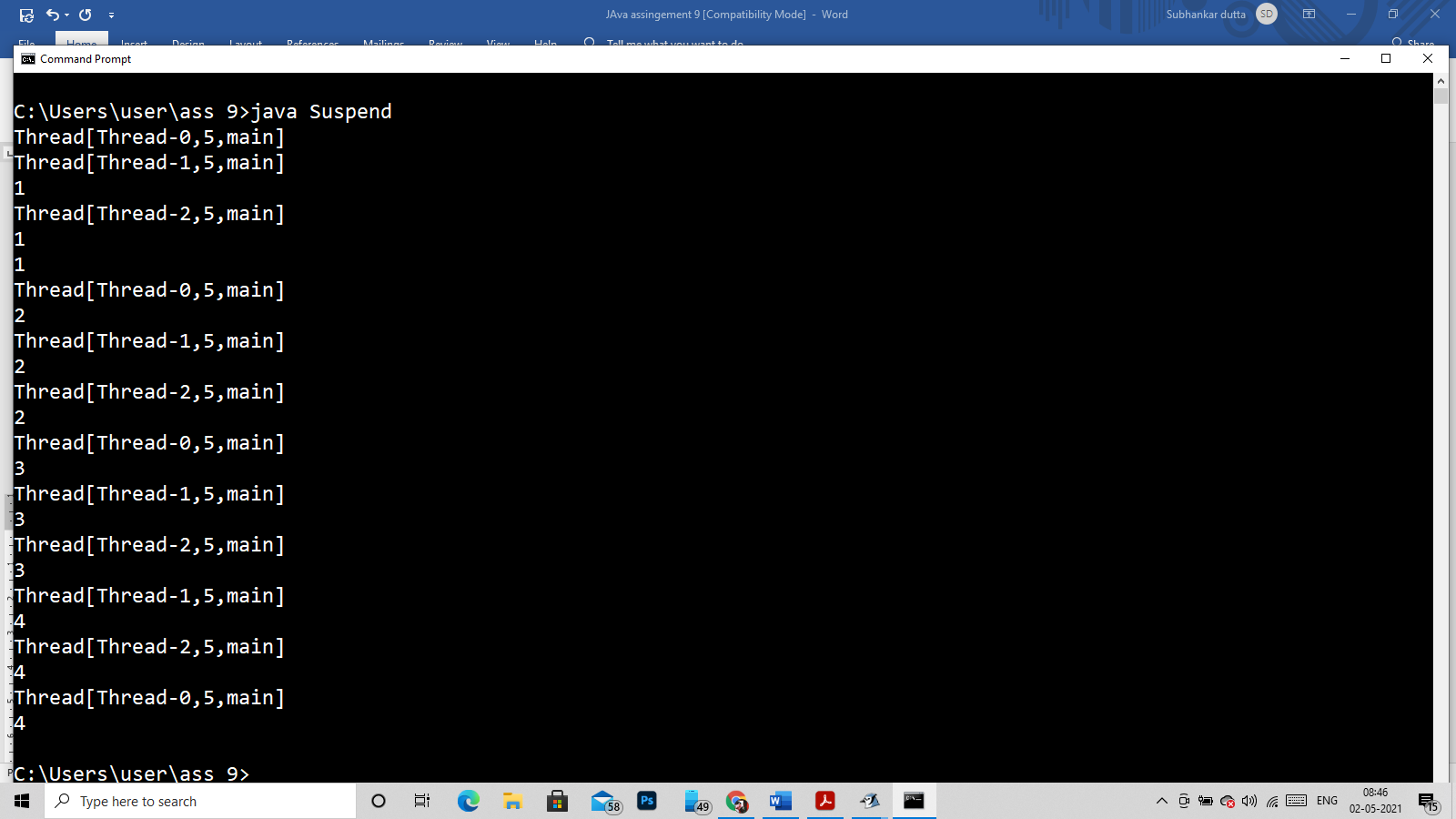
t1.start();

t2.start();

t3.start();

}

}



Q28)

class Group implements Runnable {

public void run()

{

System.out.println("UEM");

}

}

public class MainGroup

{

public static void main(String[] args)

{

ThreadGroup group1 = new ThreadGroup("Group--1");

ThreadGroup group2 = new ThreadGroup(group1, "Group--2");

Thread t1 = new Thread(group1, new Group(), "Thread-1");

Thread t2 = new Thread(group2, new Group(), "Thread-2");

t1.start();

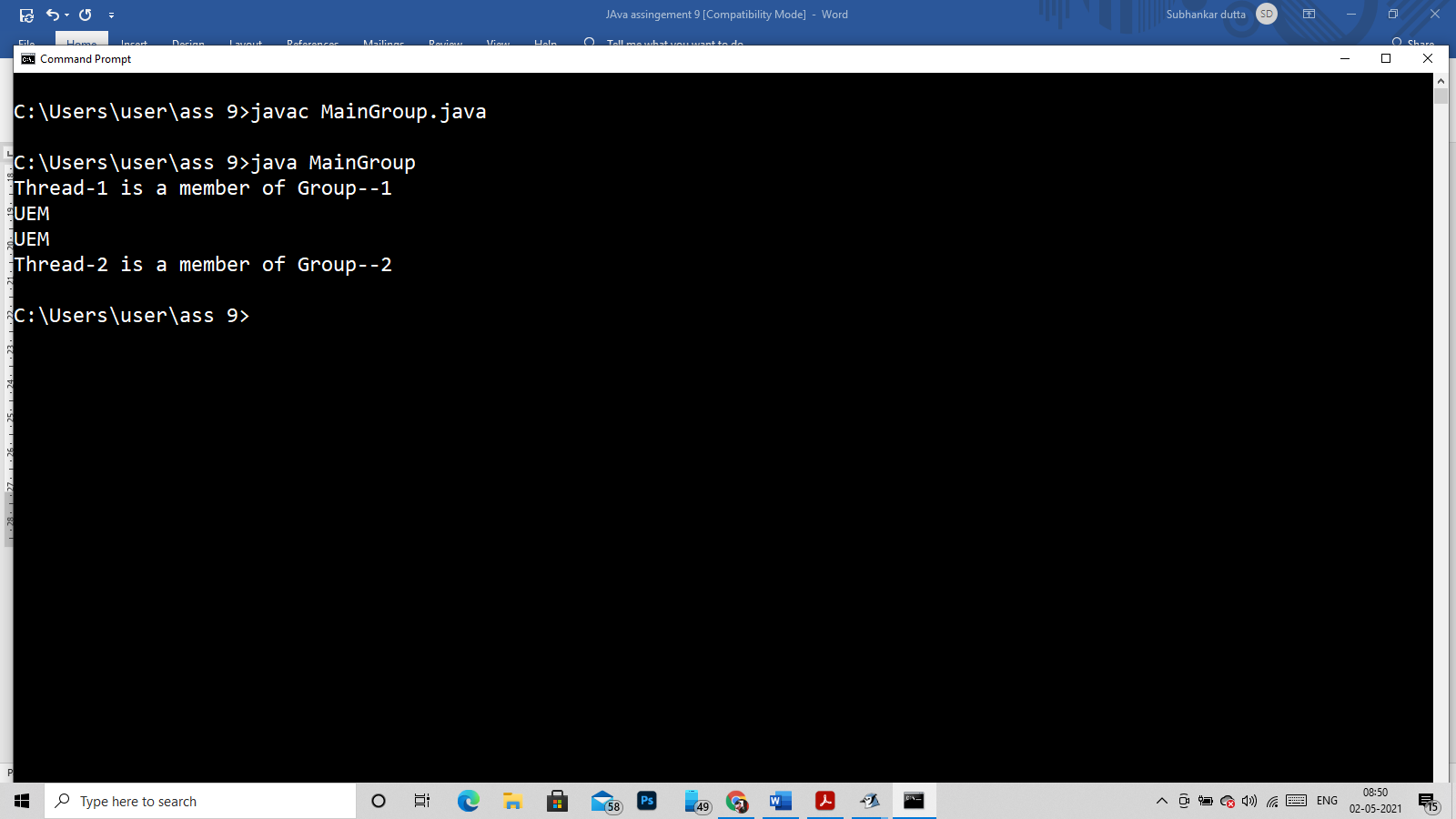
t2.start();

System.out.println(t1.getName() +" is a member of " + t1.getThreadGroup().getName());

System.out.println(t2.getName()+ " is a member of "+ t2.getThreadGroup().getName());

}

}



Q27)

class Group extends Thread {

public void run()

{

System.out.println("Thread name->> "+Thread.currentThread().getName());

}

}

public class MainGroup

{

public static void main(String[] args)

{

Group t = new Group();

Thread g = new Thread(t);

System.out.println("Calling Run() Method---- ");

//calling method run();

g.run();

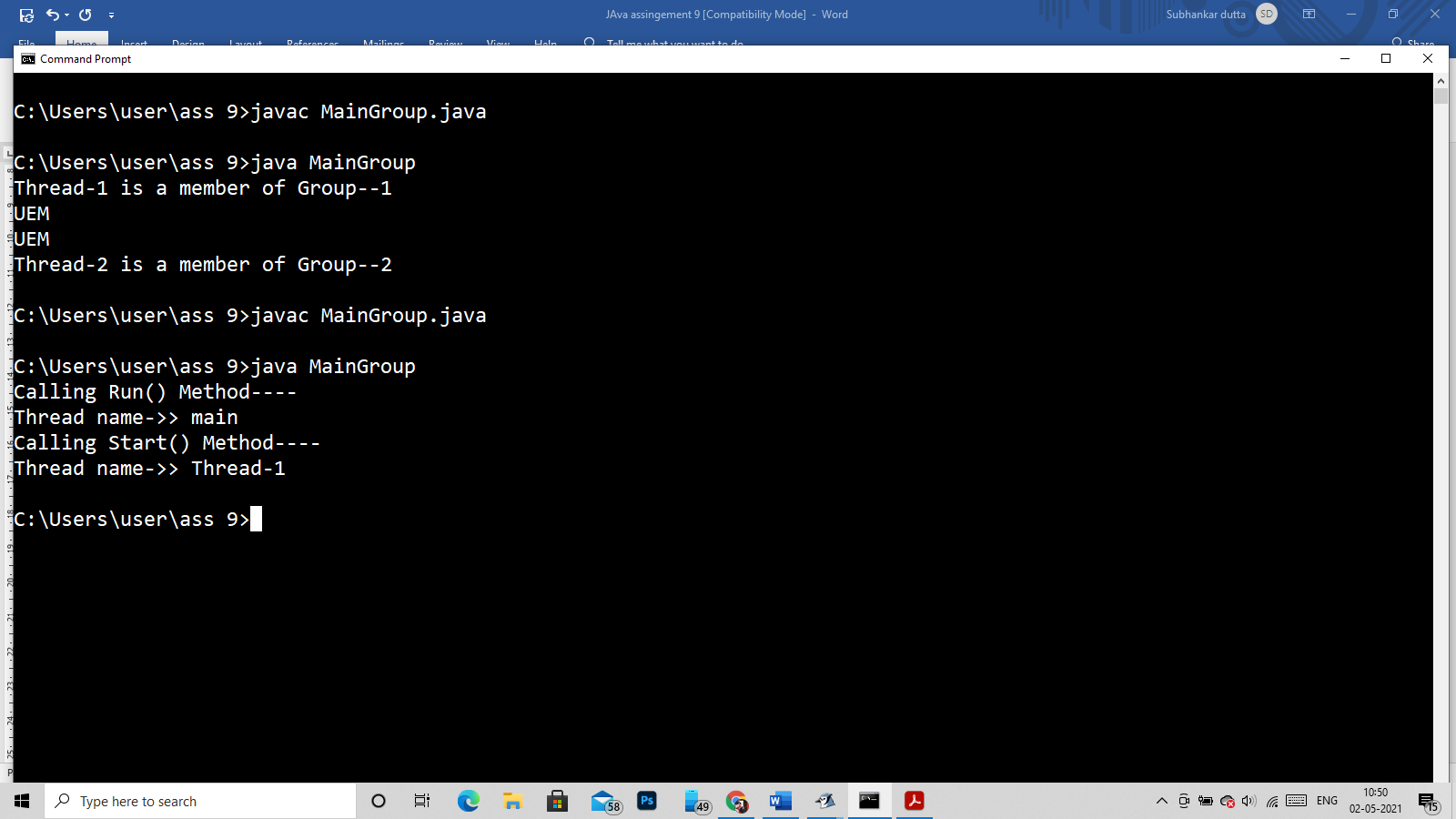
//now calling start();

System.out.println("Calling Start() Method---- ");

g.start();

}

}



Q29)

class Group extends Thread {

public void run()

{

for(int i=1;i<=5;i++)

{

System.out.println(i);

}

}

}

public class MainGroup

{

public static void main(String[] args)

{

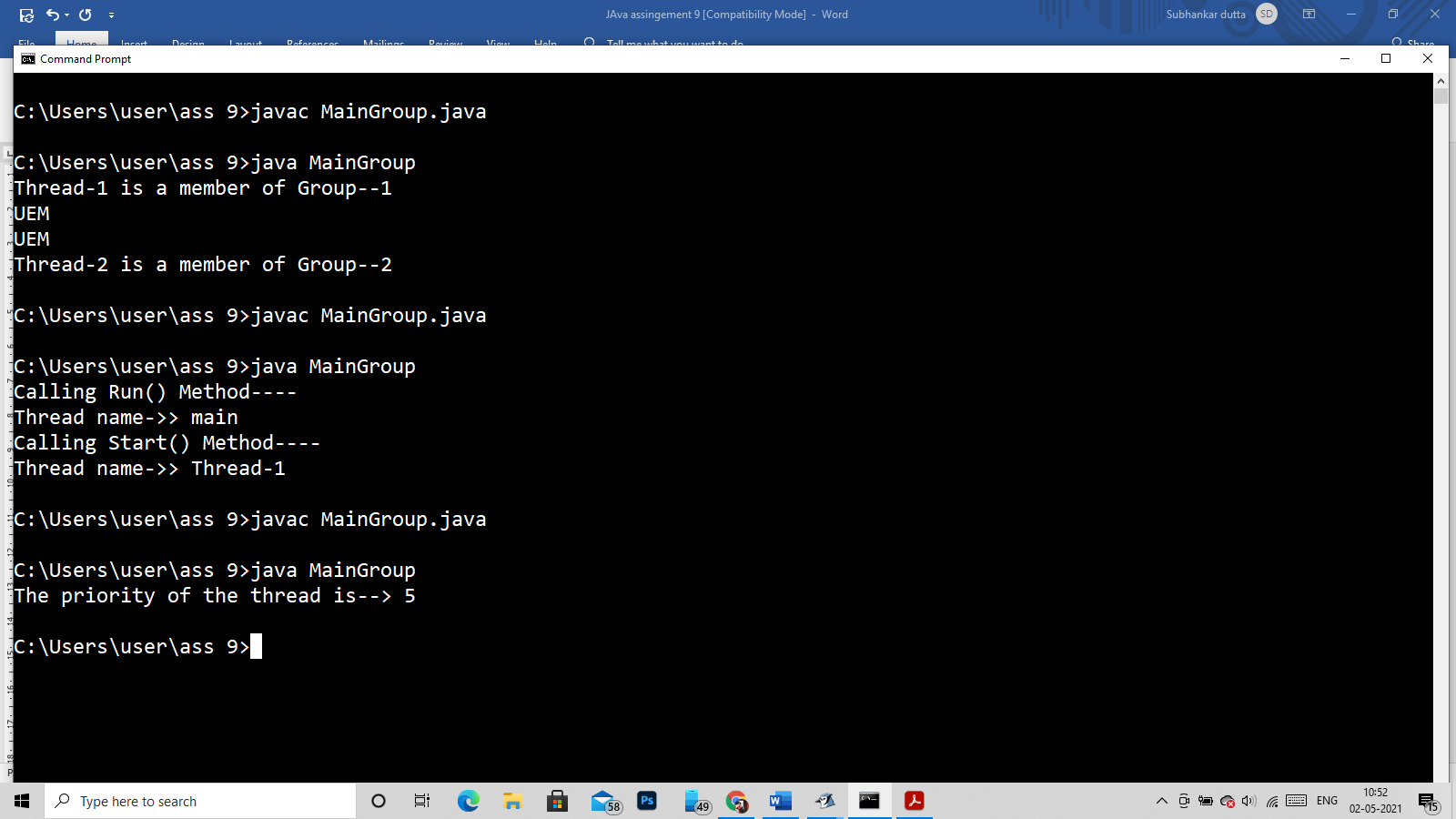
Group t = new Group();

Thread g = new Thread(t);

System.out.println("The priority of the thread is--> "+g.getPriority());

}

}



Q30)

class Group extends Thread {

public void run()

{

for(int i=1;i<=5;i++)

{

System.out.println(i);

}

}

}

public class MainGroup

{

public static void main(String[] args)

{

Group t = new Group();

Thread g = new Thread(t);

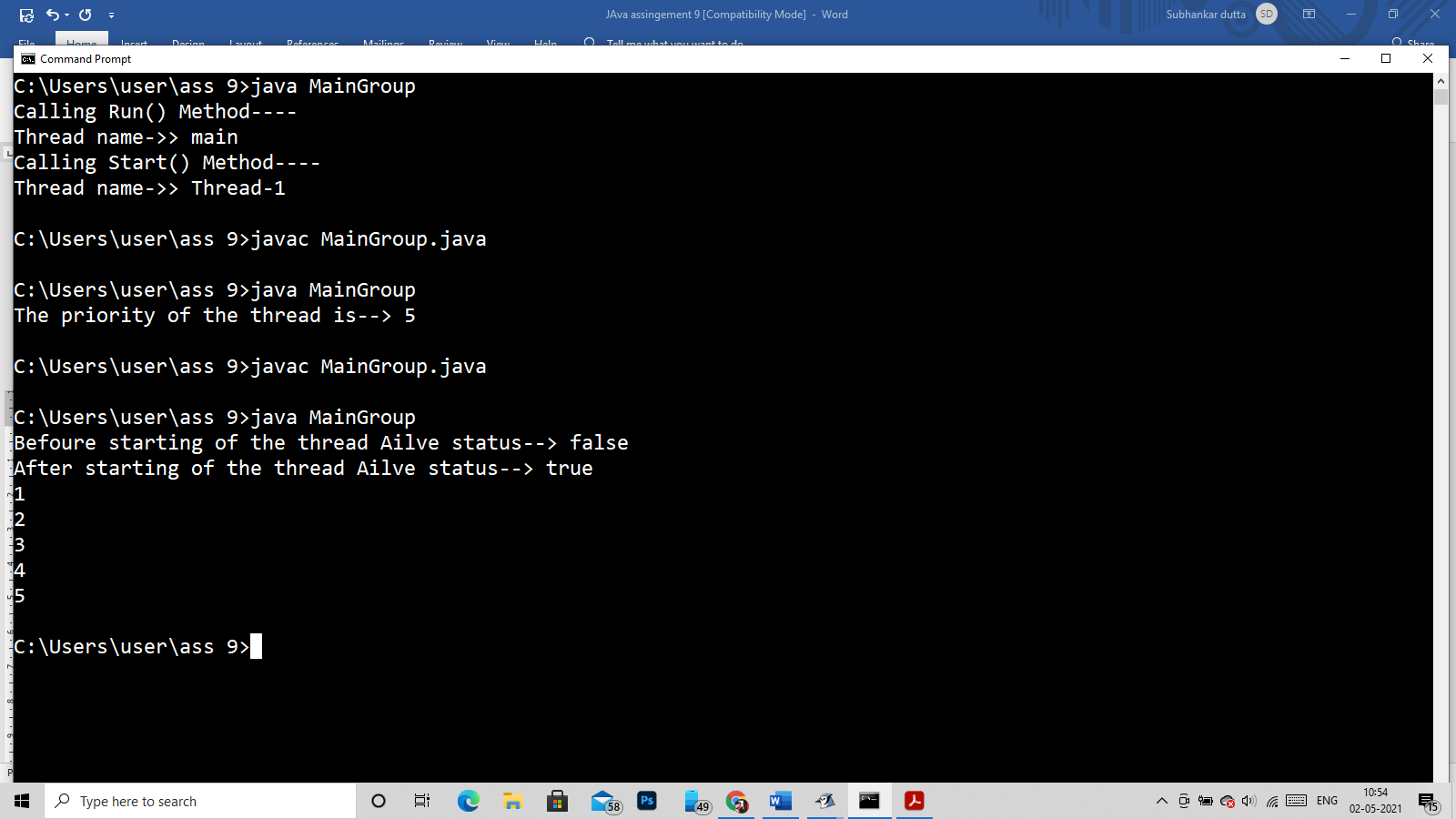
System.out.println("Befoure starting of the thread Ailve status--> "+g.isAlive());

g.start();

System.out.println("After starting of the thread Ailve status--> "+g.isAlive());

}

}



Q31)

class Group extends Thread {

public void run()

{

for(int i=1;i<=5;i++)

{

System.out.println(i);

}

}

}

public class MainGroup

{

public static void main(String[] args)

{

Group t = new Group();

Thread g = new Thread(t);

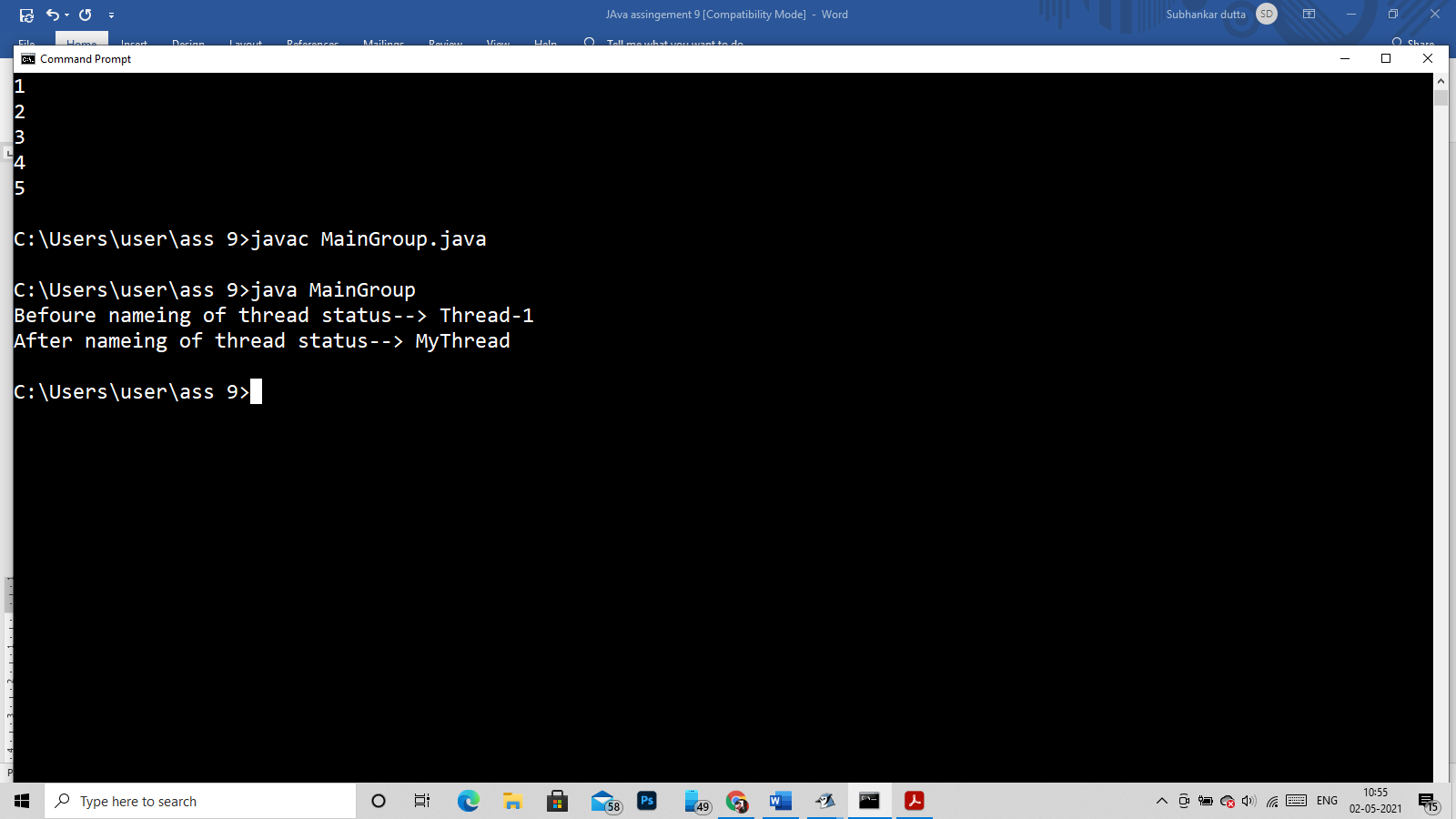
System.out.println("Befoure nameing of thread status--> "+g.getName());

g.setName("MyThread");

System.out.println("After nameing of thread status--> "+g.getName());

}

}



Q33)

class Group extends Thread {

public void run()

{

for(int i=1;i<=5;i++)

{

System.out.println("Thread Alive Status--->>>"+Thread.currentThread().isAlive());

}

}

}

public class MainGroup

{

public static void main(String[] args)

{

Group t = new Group();

Thread g = new Thread(t);

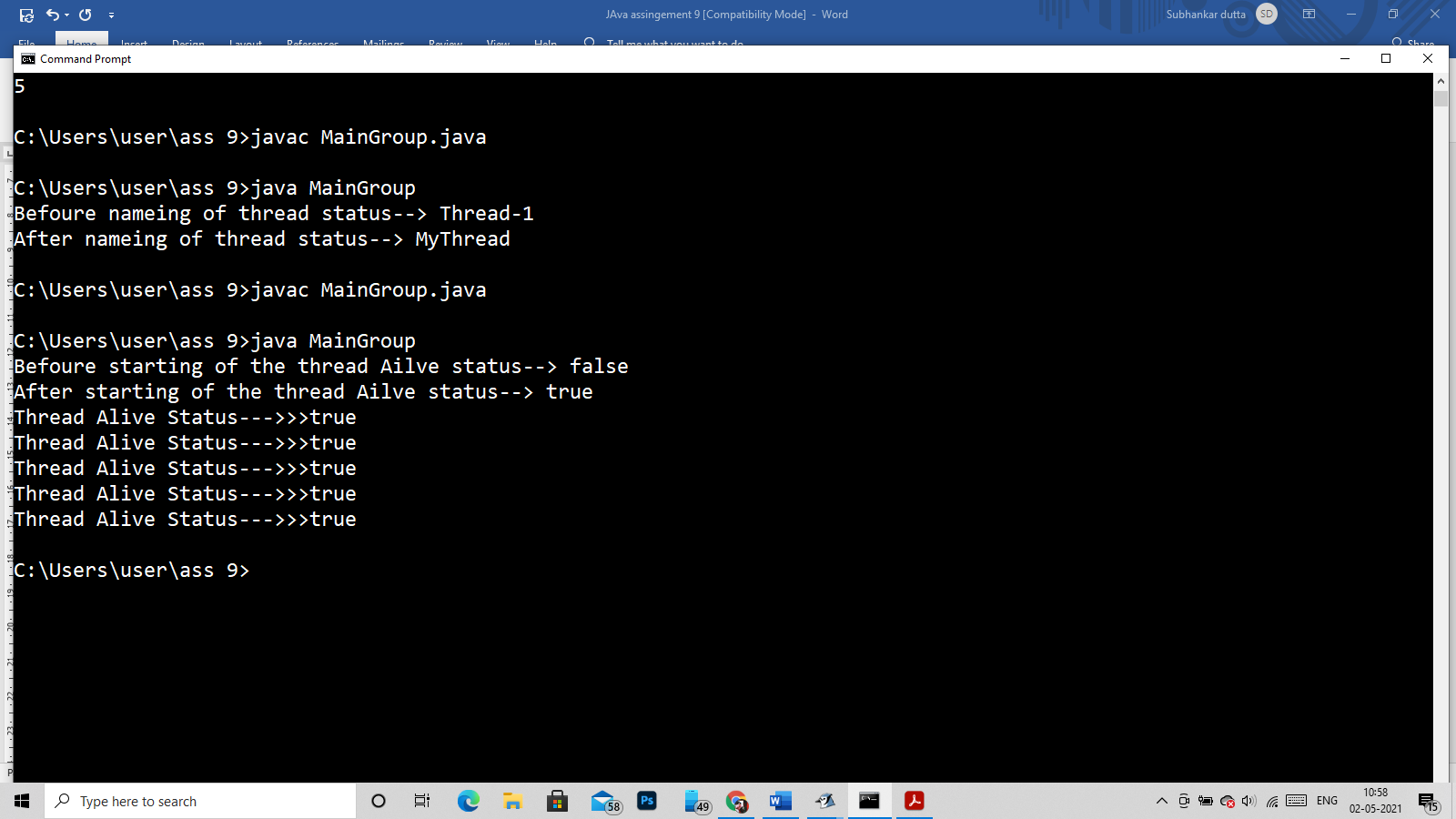
System.out.println("Befoure starting of the thread Ailve status--> "+g.isAlive());

g.start();

System.out.println("After starting of the thread Ailve status--> "+g.isAlive());

}

}



Q32)

class parentClass implements Runnable

{

public void run()

{

System.out.println("Hello i am in parentClass");

}

}

class Group extends parentClass {

void call()

{

super.run();

}

}

public class MainGroup

{

public static void main(String[] args)

{

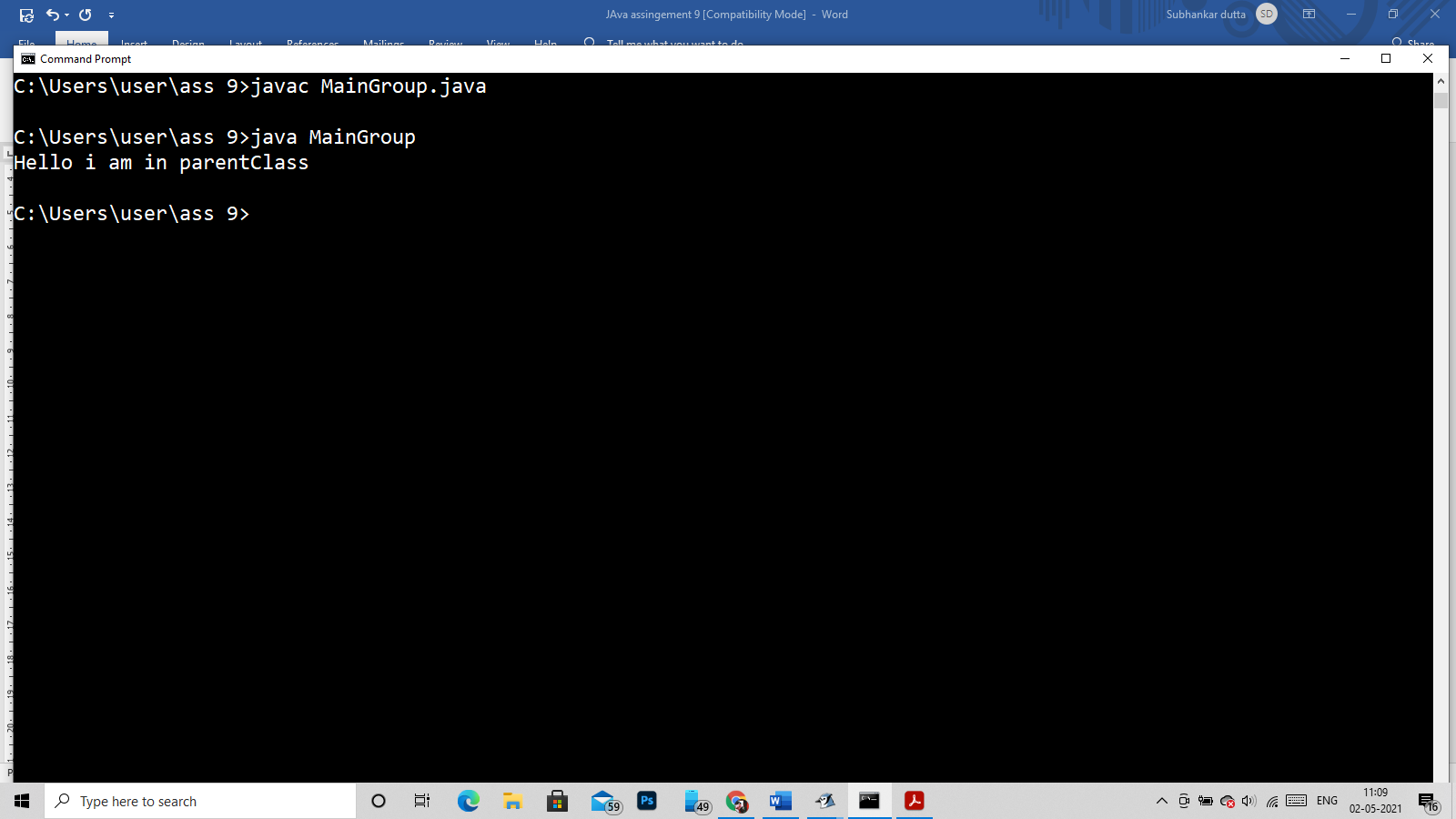
Group t = new Group();

Thread g = new Thread(t);

g.start();

}

}



Q34, Q35)

class Group extends Thread {

public synchronized void run()

{

System.out.println("==== Starting Thread--> "+Thread.currentThread().getName()+" =====");

for(int i=1;i<=5;i++)

{

System.out.println(i);

}

System.out.println("==== Ending Thread--> "+Thread.currentThread().getName()+" =====");

}

}

public class MainGroup

{

public static void main(String[] args)

{

Group g = new Group();

Thread t1 = new Thread(g);

Thread t2 = new Thread(g);

Thread t3 = new Thread(g);

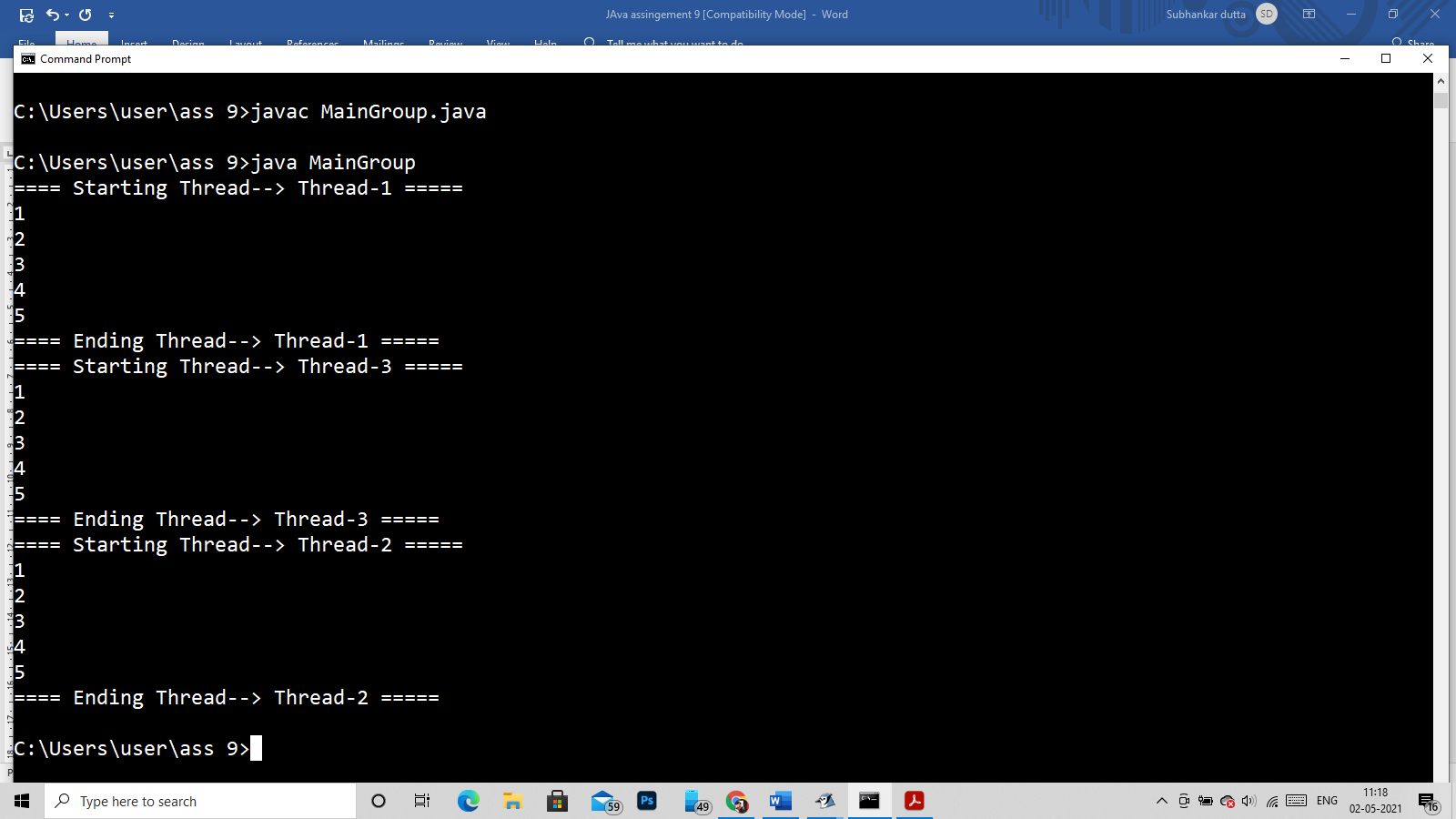
t1.start();

t2.start();

t3.start();

}

}



Q36)

If we want to pass some Args in run() method we can’t do that directly like

class Group extends Thread {

public void run(char ch)

{

System.out.println("Variable i get--> "+ch);

for(int i=1;i<=5;i++)

{

System.out.println(i);

}

}

}

public class MainGroup

{

public static void main(String[] args)

{

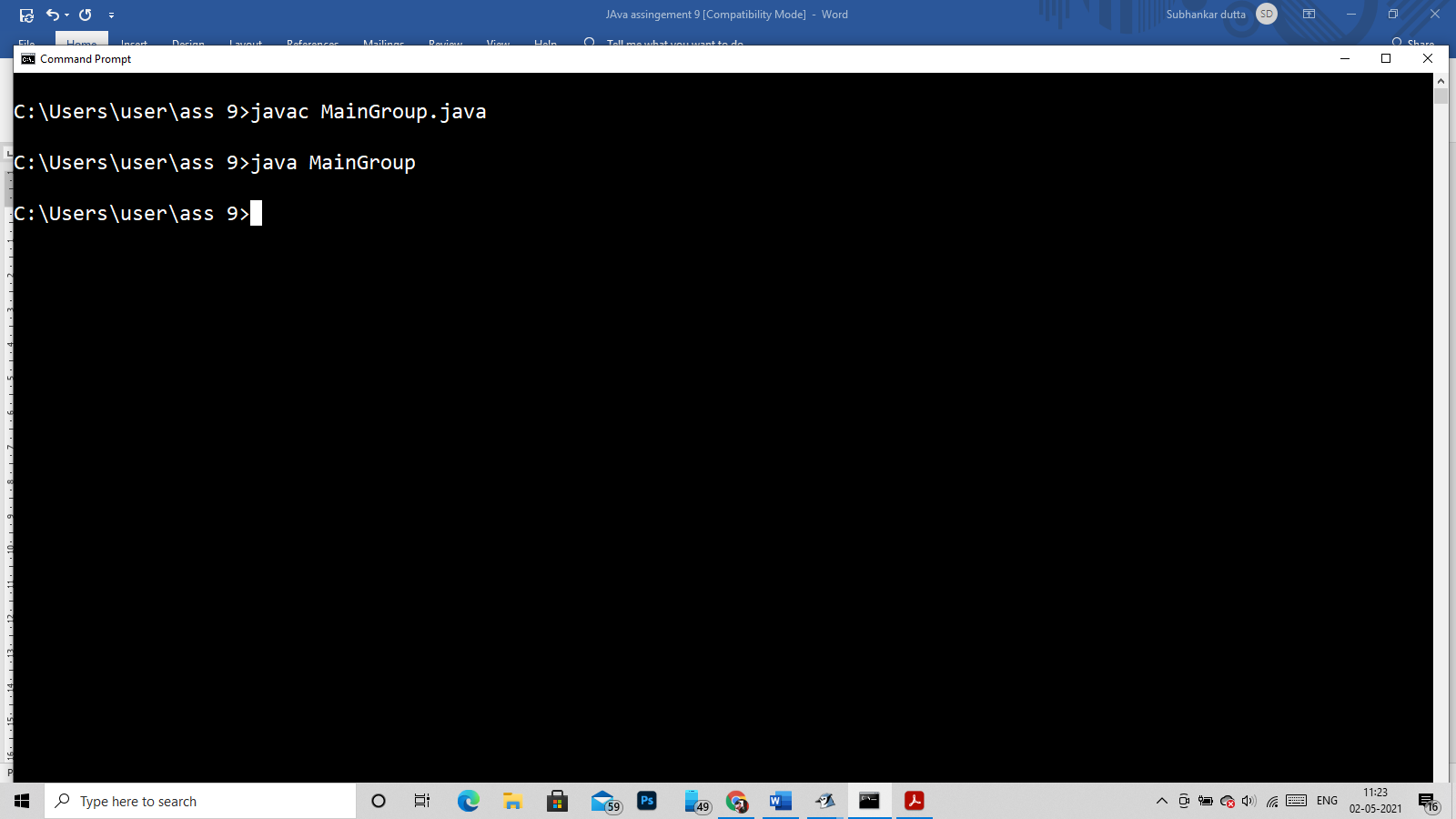
Group g = new Group();

Thread t1 = new Thread(g);

t1.start();

}

}



Compiled and Run but no Output

So for passing some Args we have to pass in the counstroctor of the class

Like->

class Group extends Thread {

String name;

Group(String name)

{

this.name=name;

}

public void run()

{

System.out.println("Variable i get--> "+name);

for(int i=1;i<=5;i++)

{

System.out.println(i);

}

}

}

public class MainGroup

{

public static void main(String[] args)

{

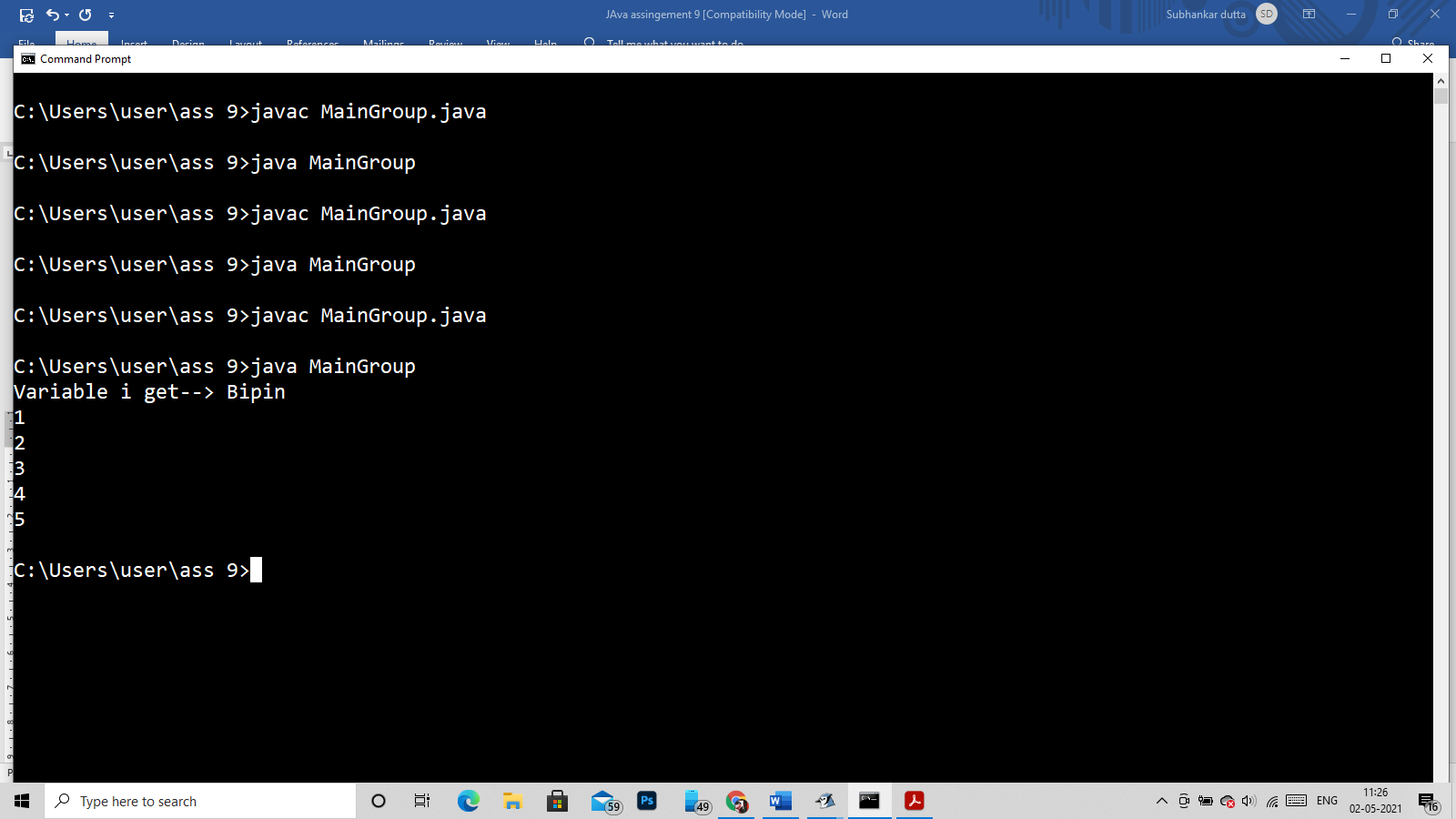
Group g = new Group("Bipin");

Thread t1 = new Thread(g);

t1.start();

}

}



Q37)

class Group extends Thread {

String name;

Group(String name)

{

this.name=name;

}

public void Summing()

{

System.out.println("Variable i get--> "+name);

for(int i=1;i<=5;i++)

{

System.out.println(i);

}

}

}

public class MainGroup

{

public static void main(String[] args)

{

Group g = new Group("Bipin");

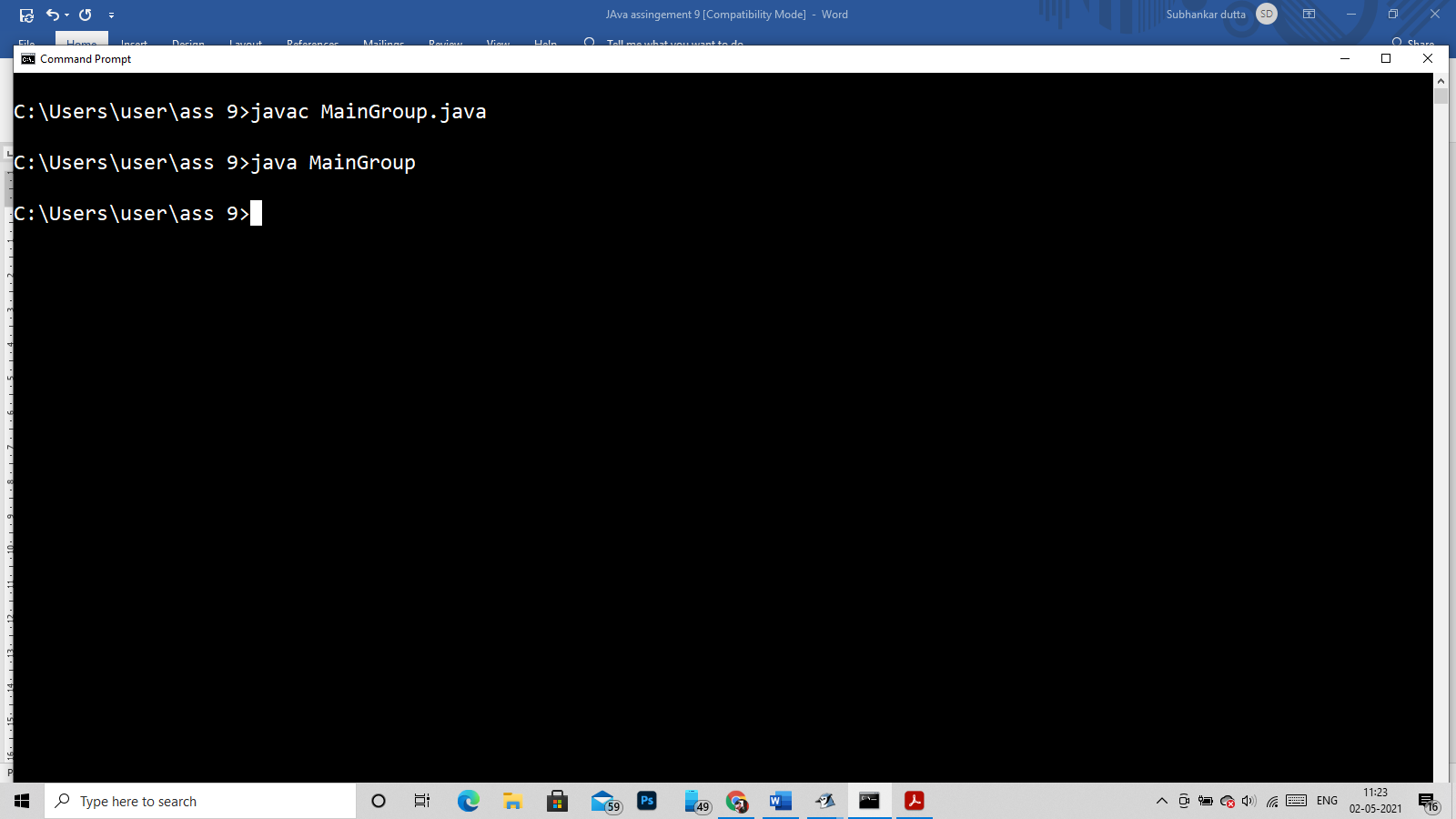
Thread t1 = new Thread(g);

t1.start();

}

}

//No output from start



Q38)

class Group extends Thread {

public void run()

{

for(int i=1;i<=10;i++)

{

System.out.println(i);

if(i==6)

{

System.out.println("Stopping Thread............");

Thread.currentThread().stop();

}

}

}

}

public class MainGroup

{

public static void main(String[] args)

{

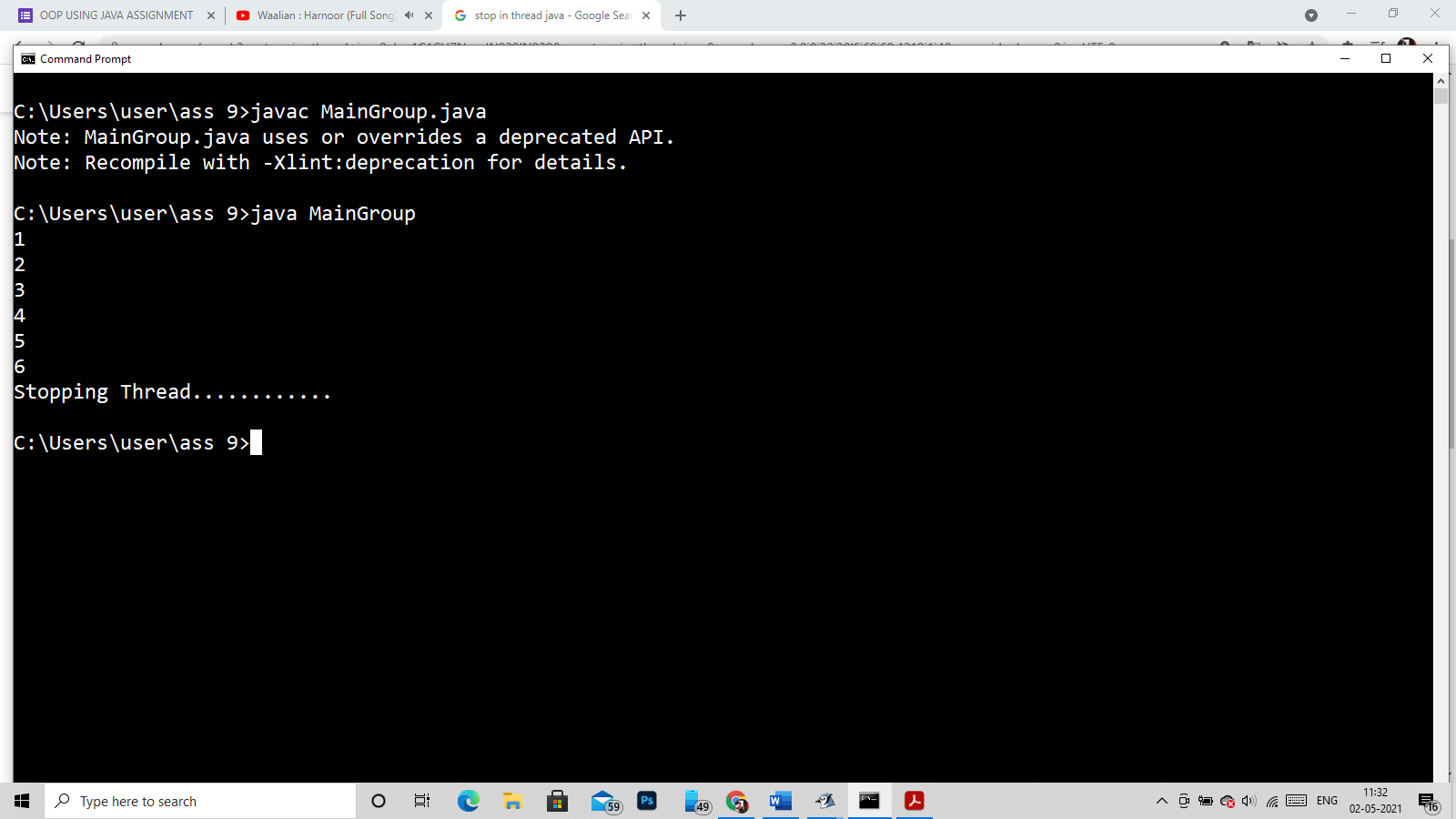
Group g = new Group();

Thread t1 = new Thread(g);

t1.start();

}

}



Q39)

class Group extends Thread {

public void run()

{

for(int i=1;i<=10;i++)

{

System.out.println(i);

}

}

}

public class MainGroup

{

public static void main(String[] args)

{

Group g = new Group();

Thread t1 = new Thread(g);

System.out.println("Wait 5 second thread is suspended......");

t1.start();

t1.suspend();

try

{

Thread.currentThread().sleep(5000);

}

catch(Exception e)

{

System.out.println(e);

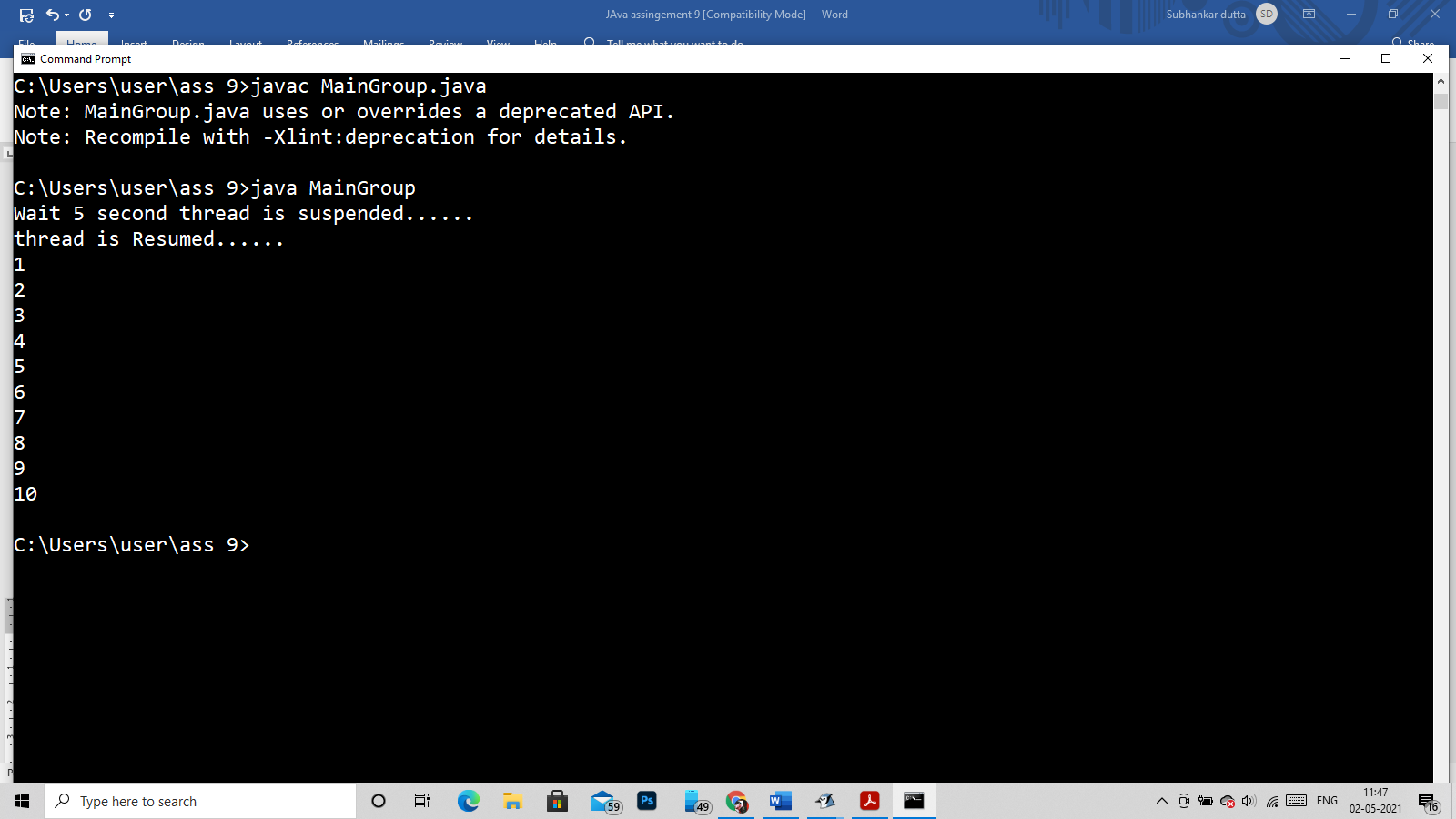
}

System.out.println("thread is Resumed......");

t1.resume();

}

}



Q40)

class Group extends Thread {

public synchronized void run()

{

for(int i=1;i<=10;i++)

{

System.out.println(i);

if(i==6)

{

System.out.println("Current Thread state::->"+Thread.currentThread().getState());

}

}

}

}

public class MainGroup

{

public static void main(String[] args)

{

Group g = new Group();

Thread t1 = new Thread(g);

System.out.println("Current Status:: "+t1.getState());

t1.start();

try

{

t1.join(100);

}catch(Exception e)

{}

System.out.println("Current Status:: "+t1.getState());

System.out.println("isAlive? Status:: "+t1.isAlive());

}

}

