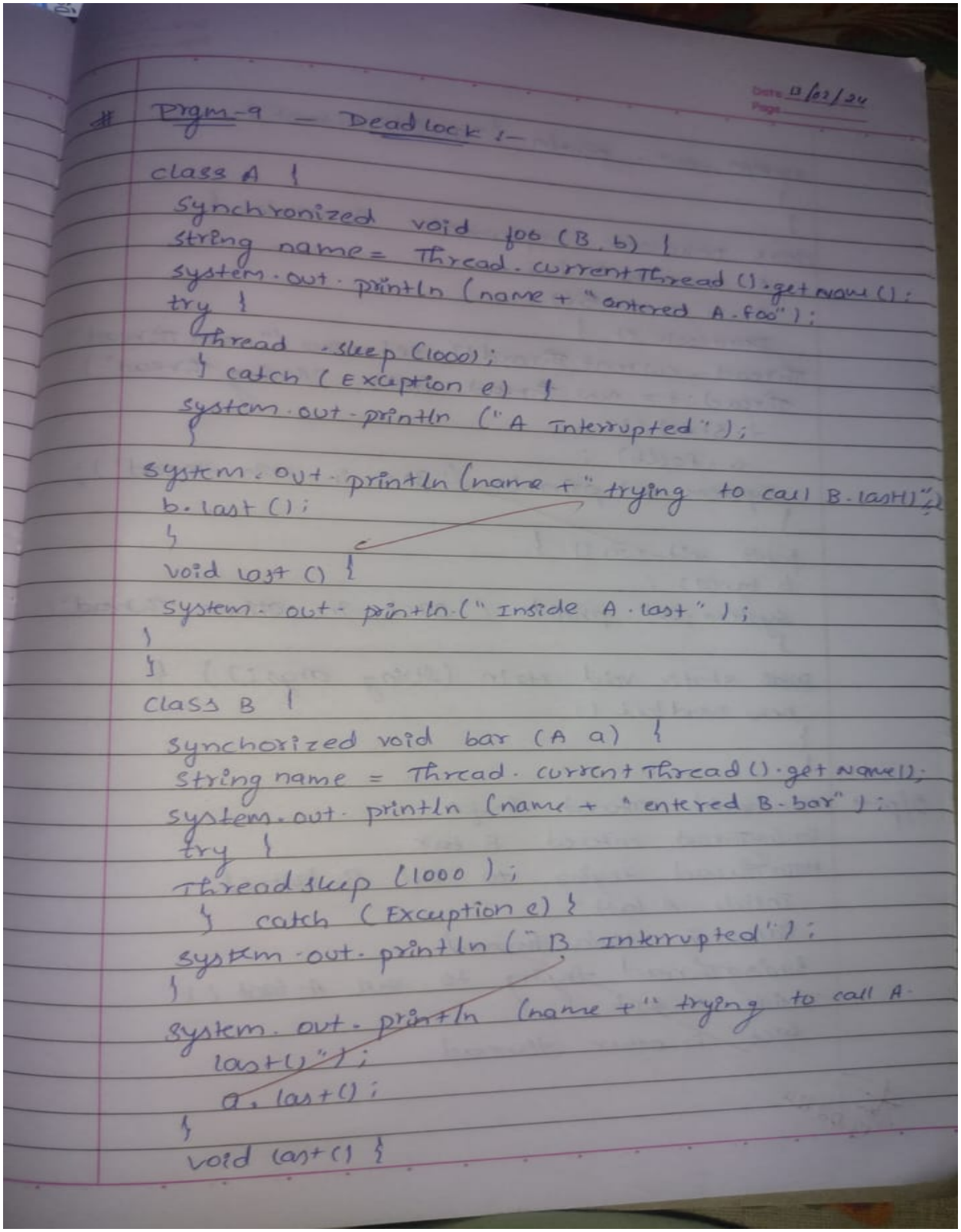
Lab 10 : deadlock



```
Date 11/03/24
Page

# Prgm-9 – Deadlock :-

class A {
 synchronized void foo (B b) {
 String name = Thread.currentThread().getName();
 system.out.println (name + "entered A.foo");
 try {
  Thread.sleep (1000);
  } catch (Exception e) {
  system.out.println ("A Interrupted");
  }
system.out.println (name + " trying to call B.lastH)";
 b.last ();
 }
 void last () {
 system.out.println (" Inside A.last ");
}
}
class B {
 synchorized void bar (A a) {
 String name = Thread.currentThread().getName();
 system.out.println (name + " entered B.bar");
 try {
 Thread.sleep (1000);
  } catch (Exception e) {
 system.out.println (" B Interrupted");
}
system.out.println (name + " trying to call A.
  last()");
  a.last();
}
 void last() {
```

```
    System.out.println("Inside A.last");
  }
}
class Deadlock implements Runnable {
  A a = new A();
  B b = new B();
  Deadlock() {
    Thread.currentThread().setName("main thread");
    Thread t = new Thread(this, "Racing thread");
    t.start();
    a.foo(b);
    System.out.println("Back in main thread");
  }
  public void run() {
    b.bar(a);
    System.out.println("Back in other thread");
  }
  public static void main(String args[]) {
    new Deadlock();
  }
}
```

o/p → Mainthread entered A.foo
Racingthread entered B.bar
Mainthread trying to call B.last()
Inside A last
Back in Main thread
RacingThread trying to call A.last()
Inside A.last
Back in other thread

Code :

```
class A {

synchronized void foo(B b) {

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

System.out.println(name + " entered
A.foo");

try {

Thread.sleep(1000);

} catch(Exception e) {

System.out.println("A Interrupted");

}
System.out.println(name + " trying to
call B.last()");

b.last();

}

void last() {

System.out.println("Inside A.last");

}

}
class B {

synchronized void bar(A a) {

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

System.out.println(name + " entered
B.bar");
```

```
try {

Thread.sleep(1000);

} catch(Exception e) {

System.out.println("B Interrupted");
}
System.out.println(name + " trying to
call A.last()");

a.last();

}
void last() {

System.out.println("Inside A.last");

}

}

class Deadlock implements Runnable
{

A a = new A();

B b = new B();

Deadlock() {
Thread.currentThread().setName("M
ainThread");

Thread t = new Thread(this,
"RacingThread");

t.start();

a.foo(b); // get lock on a in this
thread.

System.out.println("Back in main
thread");
```

```
}
public void run() {

b.bar(a); // get lock on b in other
thread.

System.out.println("Back in other
thread");

}

public static void main(String args[]) {

new Deadlock();

}

}
```

Output :