R. CONCEICAO RODRIGUES COLLEGE OF ENGINEERIG
Department of Computer Engineering

Experiment 9- Based on Multithreading

1. **Course Details:**

Academic Year	2023 - 24	Estimated Time	Experiment No.9 – 02 Hours
Course & Semester	S.E. (COMP) – Sem. III	Subject Name	Skill based lab Course-OOP with Java
Module No.	05	Chapter Title	Exception Handling and multithreading
Experiment Type	Software Performance	Subject Code	CSL304

Name of Student	Mark Lopes	Roll No.	9913
Date of Performance:	13/10/23	Date of Submission:	20/10/23
CO Mapping	**CSL304.4 Implement the concept of inheritance, exception handling and multithreading**		

Timeline (2)	Preparedness (2)	Effort (3)	Result (3)	Total (10)

Problem statement:

1) Write a java program to create the child thread, comment on the execution of main and child thread.

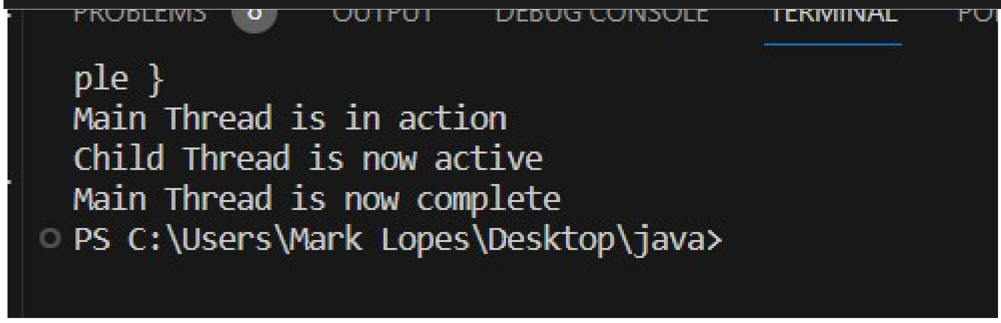
2) Using above example demonstrate the following methods.
sleep(), join(), getPrioity(), setPriority(), getName(), setName(),getid(),currentThread(), yield(), suspend(), resume().

3) Simulate the simultaneous transactions on ‘withdraw’ and ‘deposit’ on bank account. Demonstrate using multithreading.

Q1

```
class ChildThread extends Thread {

    public void run() {

        System.out.println("Child Thread is now active");

    }

}


public class ThreadExecutionExample {

    public static void main(String[] args) {

        // Create a child thread

        ChildThread childThread = new ChildThread();


        // Launch the child thread

        childThread.start();


        // Main thread's journey begins

        System.out.println("Main Thread is in action");


        // Optionally, wait for the child thread to finish

        try {

            childThread.join();

        } catch (InterruptedException e) {

            System.out.println("Main Thread was interrupted while
waiting for the child thread.");

        }
```

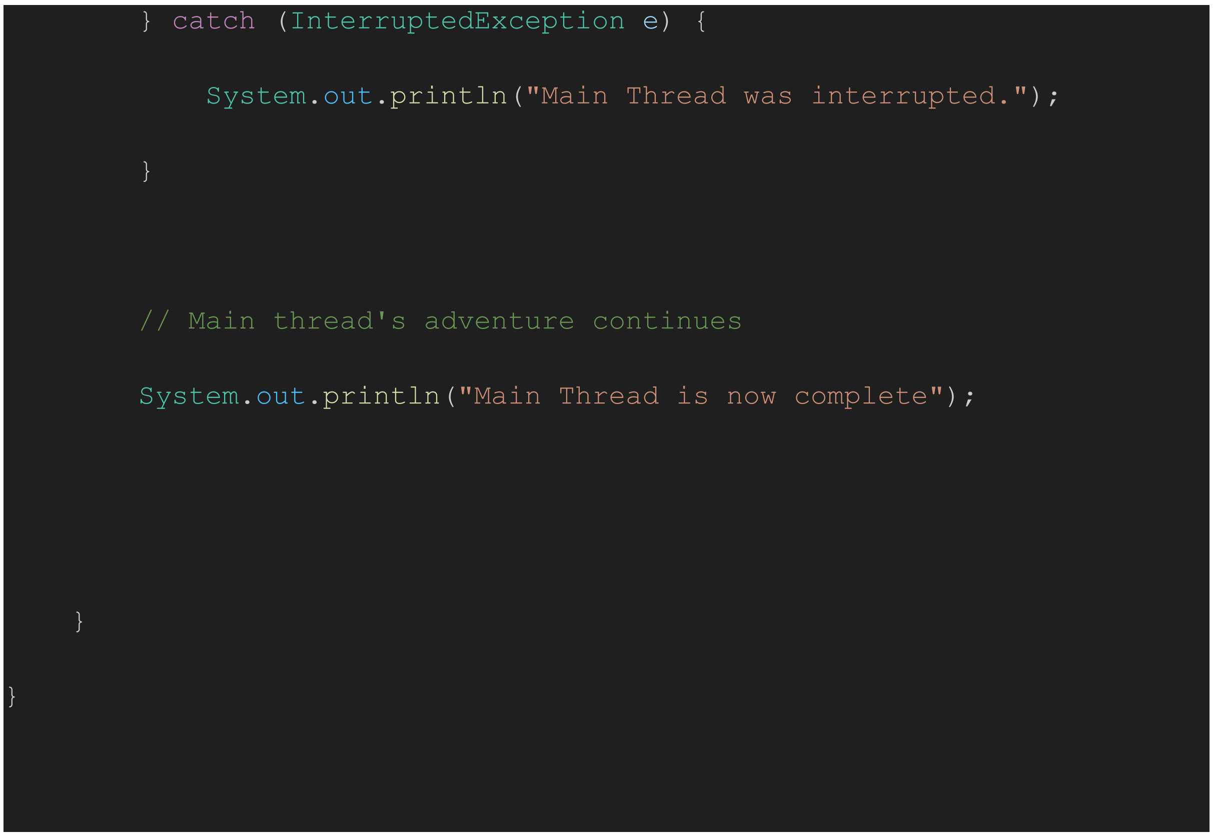
```java
        // Main thread's adventure continues

        System.out.println("Main Thread is now complete");

    }

}
```



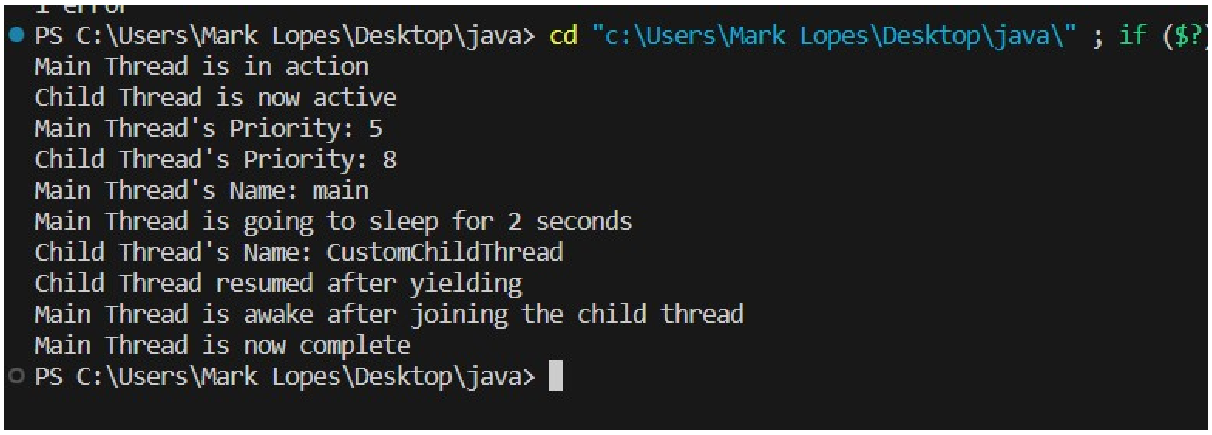
```
ple }
Main Thread is in action
Child Thread is now active
Main Thread is now complete
PS C:\Users\Mark Lopes\Desktop\java>
```

Q2

```java
class MyChildThread extends Thread {

    public void run() {

        System.out.println("Child Thread is now active");

        System.out.println("Child Thread's Priority: " +
getPriority());

        System.out.println("Child Thread's Name: " + getName());

        Thread.yield(); // Yield to another thread

        System.out.println("Child Thread resumed after yielding");

    }

}



public class ThreadMethodsDemo {

    public static void main(String[] args) {

        // Create a custom child thread
        MyChildThread childThread = new MyChildThread();


        // Set thread priority and name for the child thread

        childThread.setPriority(8);

        childThread.setName("CustomChildThread");


        // Launch the child thread

        childThread.start();


        // Main thread's journey begins

        Thread mainThread = Thread.currentThread();

        System.out.println("Main Thread is in action");

        System.out.println("Main Thread's Priority: " +
mainThread.getPriority());

        System.out.println("Main Thread's Name: " +
mainThread.getName());


        try {

            // Main thread sleeps for 2 seconds

            System.out.println("Main Thread is going to sleep for 2
seconds");

            Thread.sleep(2000);


            // Main thread joins the child thread

            childThread.join();

            System.out.println("Main Thread is awake after joining the
child thread");
```



```java
        } catch (InterruptedException e) {

            System.out.println("Main Thread was interrupted.");

        }



        // Main thread's adventure continues

        System.out.println("Main Thread is now complete");




    }

}
```

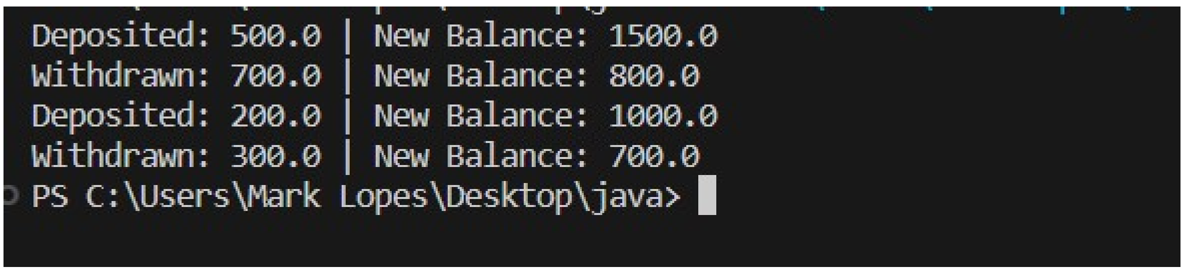


```
● PS C:\Users\Mark Lopes\Desktop\java> cd "c:\Users\Mark Lopes\Desktop\java\" ; if ($?
  Main Thread is in action
  Child Thread is now active
  Main Thread's Priority: 5
  Child Thread's Priority: 8
  Main Thread's Name: main
  Main Thread is going to sleep for 2 seconds
  Child Thread's Name: CustomChildThread
  Child Thread resumed after yielding
  Main Thread is awake after joining the child thread
  Main Thread is now complete
○ PS C:\Users\Mark Lopes\Desktop\java> █
```

Q3

```java
class BankAccount {
    private double balance;

    public BankAccount(double initialBalance) {
        // Initialize the bank account with an initial balance.
        balance = initialBalance;
    }

    public synchronized void deposit(double amount) {
        // Deposit money into the account and show the new balance.
        balance += amount;
        System.out.println("Deposited: " + amount + " | New Balance: "
+ balance);
    }

    public synchronized void withdraw(double amount) {
        if (balance >= amount) {
            // If there's enough balance, withdraw money and show the
new balance.
            balance -= amount;
            System.out.println("Withdrawn: " + amount + " | New
Balance: " + balance);
        } else {
            // If there's not enough balance, show a message.
            System.out.println("Insufficient balance to withdraw " +
amount);
```

```java
        }
    }
}

class DepositThread extends Thread {
    private BankAccount account;
    private double amount;

    public DepositThread(BankAccount account, double amount) {
        this.account = account;
        this.amount = amount;
    }

    public void run() {
        // Perform a deposit transaction.
        account.deposit(amount);
    }
}

class WithdrawThread extends Thread {
    private BankAccount account;
    private double amount;

    public WithdrawThread(BankAccount account, double amount) {
        this.account = account;
        this.amount = amount;

    }


    public void run() {

        // Perform a withdrawal transaction.

        account.withdraw(amount);

    }

}


public class BankTransactionDemo {

    public static void main(String[] args) {

        // Create a bank account with an initial balance of $1000.

        BankAccount account = new BankAccount(1000.0);



        // Create threads to simulate transactions.

        DepositThread depositThread1 = new DepositThread(account,
200.0);

        WithdrawThread withdrawThread1 = new WithdrawThread(account,
300.0);

        DepositThread depositThread2 = new DepositThread(account,
500.0);

        WithdrawThread withdrawThread2 = new WithdrawThread(account,
700.0);



        // Start the threads to perform transactions.

        depositThread1.start();

        withdrawThread1.start();
        depositThread2.start();

        withdrawThread2.start();

    }

}
```



```
Deposited: 500.0 | New Balance: 1500.0
Withdrawn: 700.0 | New Balance: 800.0
Deposited: 200.0 | New Balance: 1000.0
Withdrawn: 300.0 | New Balance: 700.0
PS C:\Users\Mark Lopes\Desktop\java>
```