**Assignment Week 3 Multithreading**

**Program 1: Create a thread by implementing runnable interface**

public class \_01\_Thread\_using\_Runnable {

    public static void main(String[] args) {

        NewThread nt = new NewThread();

        Thread thread=new Thread(nt,"number thread");

        thread.start();

    }

    private static class NewThread implements Runnable {

        @Override

        public void run() {

            for(int i=0;i<10;i++){

                System.out.println("Child Thread: "+i);

                try{

                    Thread.sleep(1000);

                }

                catch(InterruptedException e){

                    e.printStackTrace();

                }

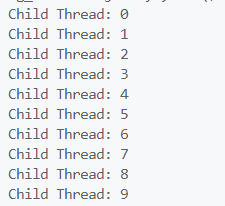
            }

        }

    }

}

**Output-**

****

**Program 2: Print even and odd numbers using different threads**

 class PrintEvenThread implements Runnable {

    private int maxVal;

    public PrintEvenThread(int maxVal) {

        this.maxVal = maxVal;

    }

    @Override

    public void run() {

        for (int i = 2; i <= maxVal; i += 2) {

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

            try {

                Thread.sleep(100);

            } catch (InterruptedException e) {

                e.printStackTrace();

            }

        }

    }

}

class PrintOddThread implements Runnable {

    private int maxVal;

    public PrintOddThread(int maxVal) {

        this.maxVal = maxVal;

    }

    @Override

    public void run() {

        for (int i = 1; i <= maxVal; i += 2) {

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

            try {

                Thread.sleep(100);

            } catch (InterruptedException e) {

                e.printStackTrace();

            }

        }

    }

}

public class \_02\_Even\_and\_Odd\_Number {

    public static void main(String[] args) {

        int maxVal = 20;

        PrintEvenThread even = new PrintEvenThread(maxVal);

        PrintOddThread odd = new PrintOddThread(maxVal);

        Thread evenThread = new Thread(even, "Even");

        Thread oddThread = new Thread(odd, "Odd");

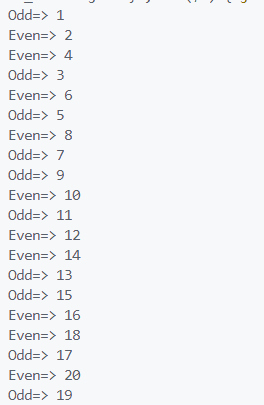
        evenThread.start();

        oddThread.start();

    }

}

**Output-**

****

**Program 3: Create one user and two daemon threads**

public class \_03\_Two\_Daemon\_and\_One\_UserThreads {

    public static void main(String[] args) {

        // Creating two daemon threads

        Thread dt1 = new Thread(new DaemonTask(), "Daemon Thread 1");

        Thread dt2 = new Thread(new DaemonTask(), "Daemon Thread 2");

        // Setting daemon status for the daemon threads

        dt1.setDaemon(true);

        dt2.setDaemon(true);

        // Creating a user thread

        Thread tUser = new Thread(new UserTask(), "User Thread 1");

        // Starting all threads

        dt1.start();

        dt2.start();

        tUser.start();

        try {

            Thread.sleep(2000);

        } catch (InterruptedException e) {

            e.printStackTrace();

        }

        System.out.println("Exiting main thread... "+Thread.currentThread().getName());

    }

    // Task for daemon threads

    private static class DaemonTask implements Runnable {

        @Override

        public void run() {

            while (true) {

                System.out.println(Thread.currentThread().getName() + " is up and running... ");

                try {

                    Thread.sleep(1000);

                } catch (InterruptedException e) {

                    e.printStackTrace();

                }

            }

        }

    }

    // Task for user thread

    private static class UserTask implements Runnable {

        @Override

        public void run() {

            System.out.println(Thread.currentThread().getName() + " is up and running...");

            try {

                Thread.sleep(3000);

            } catch (InterruptedException e) {

                e.printStackTrace();

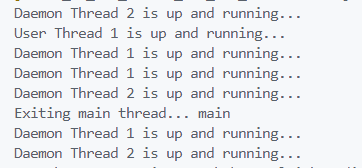
            }

        }

    }

}

**Output-**

****

**Program 4: Create a thread which contains below methods**

* **getID()**
* **isAlive()**
* **currentThread()**
* **sleep(milliseconds)**

public class \_04\_ThreadMethods {

    public static void main(String[] args) {

        CustomThread nt = new CustomThread();

        System.out.println(nt.getID());

        System.out.println(nt.isAlive());

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

        nt.sleep(1000);

    }

}

class CustomThread implements Runnable {

    private Thread thread;

    public CustomThread() {

        thread = new Thread(this);

        thread.start();

    }

    public long getID() {

        return thread.getId();

    }

    public boolean isAlive() {

        return thread.isAlive();

    }

    public Thread currentThread() {

        return Thread.currentThread();

    }

    public void sleep(long milliseconds) {

        try {

            Thread.sleep(milliseconds);

        } catch (InterruptedException e) {

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

        }

    }

    @Override

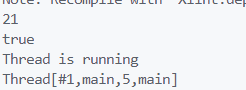
    public void run() {

        System.out.println("Thread is running");

    }

}

**Output-**

****