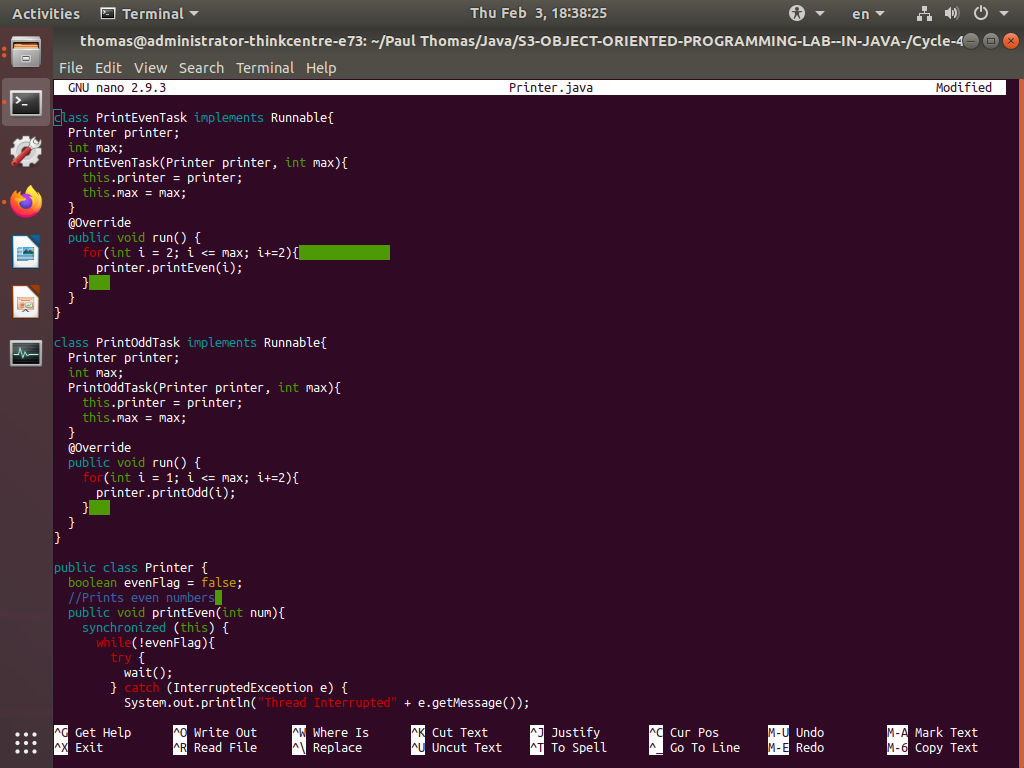
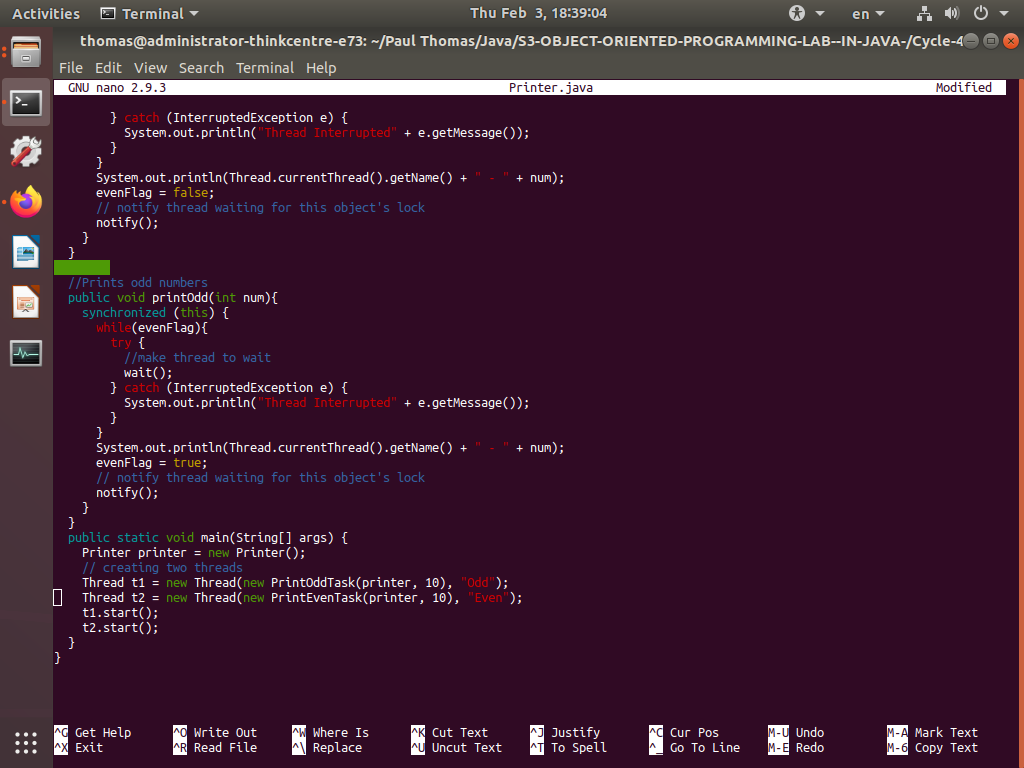
**Experiment : 4.3 Paul Thomas 30 S3 CSB**

**Program**

| class PrintEvenTask implements Runnable{  Printer printer;  int max;  PrintEvenTask(Printer printer, int max){  this.printer = printer;  this.max = max;  }  @Override  public void run() {  for(int i = 2; i <= max; i+=2){  printer.printEven(i);  }  }  }  class PrintOddTask implements Runnable{  Printer printer;  int max;  PrintOddTask(Printer printer, int max){  this.printer = printer;  this.max = max;  }  @Override  public void run() {  for(int i = 1; i <= max; i+=2){  printer.printOdd(i);  }  }  }  public class Printer {  boolean evenFlag = false;  //Prints even numbers  public void printEven(int num){  synchronized (this) {  while(!evenFlag){  try {  wait();  } catch (InterruptedException e) {  System.out.println("Thread Interrupted" + e.getMessage());  }  }  System.out.println(Thread.currentThread().getName() + " - " + num); | evenFlag = false;  // notify thread waiting for this object's lock  notify();  }  }  //Prints odd numbers  public void printOdd(int num){  synchronized (this) {  while(evenFlag){  try {  //make thread to wait  wait();  } catch (InterruptedException e) {  System.out.println("Thread Interrupted" + e.getMessage());  }  }  System.out.println(Thread.currentThread().getName() + " - " + num);  evenFlag = true;  // notify thread waiting for this object's lock  notify();  }  }  public static void main(String[] args) {  Printer printer = new Printer();  // creating two threads  Thread t1 = new Thread(new PrintOddTask(printer, 10), "Odd");  Thread t2 = new Thread(new PrintEvenTask(printer, 10), "Even");  t1.start();  t2.start();  }  } |
| --- | --- |

Program





Output

