Computational Problem Solving in Domain II Basic Study Guide

“What will be on exam?” – as according to Professor Erik Golen

Edward Riley

A close up of a mans face

Description generated with high confidence

1. **Binary IO – DataInputStream (IN doWork() method – see Thread how to call it)**
   1. Public boolean addInts(String **[FILE\_NAME]**) {
   2. DataInputStream dis = null;
   3. Try { dis = new DataInputStream(new FileInputStream**[FILE\_NAME]**) }
   4. While (true) { int integer = dis.readInt(); count++; sumRead = sumRead + integer; list.add(integer); } //Will read everything in file until reaches end of the file exception
   5. Catch (FileNotFoundException fnfe) {//code exception }
   6. Catch (EOFException eof) { Platform.runLater( new Runnable() { public void () run { taOutput.appendText(String.format(//code output)); } } ); **return true (for specific Boolean method – tells us the files found is working)**;}
   7. Catch (Exception e) { //code exception }
   8. Finally {try { dis.close(); } catch {//code exception} } //finally executes no matter what exception it is

**MUST READ THROUGH ALL FIELD IN ORDER TO GO TO THE NEXT LINE**

**CONCEPT MODEL**

**FILE**

**FIELD1 FIELD2 FIELD3 FIELD4**

**A -> B -> C -> D ->**

**-> E -> F -> G -> H -> [END OF FILE]**

1. **Threads (Practical will go straight to Threading – not simultaneously with other method)**
   1. **[METHOD 1]** Platform.runLater( new runnable() { public void run() { // code } } ) ;
   2. **[METHOD 2]** Thread t = new Thread() { run() { doWork() } };
   3. **[PROFESSOR METHOD – BEGINNING]** public class ReadInts extends Thread { public void run() { //code **(OPTIONAL)** public ReadInts(String \_fileName){ filename = \_fileName; //code constructor }} }
   4. **[CONT. PROFESSOR METHOD – PART ONE OF THREADS]** ReadInts myThread = new ReadInts(tfFileName.getText() + i + “.dat”); myThread.start(); myThread.join();
   5. **[CONT. PROFESSOR METHOD – PART TWO OF THREADS]** public void doWork() { int i = 1; ReadInts[] threads = new ReadInts[7]; (for int 1; i <= threads.length; i++) { threads[i-1] = new ReadInts(tfFileName.getText() + i + “.dat”); myThread[i-1].start(); for int (i = 1; i <= threads.length; i++) {myThread[i-1].join(); }
2. **How to catch all file & slow output?** 
   1. While (addInts()(tfFileName.getText() + i + “.dat”)){ i++ try { Thread.sleep(1000); } catch (InterruptedException ie) { // code exception }
3. **Synchronized**
   1. **[in while (true) loop]** Synchronized(list) { list.add(integer) };

SCROLL TO BOTTOM FOR PARTS OF EXAM

**PARTS OF THE EXAM**

**We will be provided with a Text Area (starter GUI) and move on.**

**CRIB SHEET ALLOWED – MUST BE HANDWRITTEN**

1. **Threads (60%)**
   1. **Write a Thread class**
   2. **Create array of Threads**
      1. **doWork();**
   3. **Run all Threads**
      1. **doWork();**
   4. **Wait for Threads to finish**
      1. **doWork();**
   5. **Write to a shared variable (TextArea(?))**
   6. **Synchronize on *SOME* ( >:( ) object**
2. **BINARY I/O (40%)**
   1. **Open Binary File [DataInputStream]**
   2. **Read the Data**
   3. **Perform Computations on the Data**
      1. **SMALLEST VALUE OF SOMETHING?!**
      2. **WHO IS THE LARGEST FOOTBALL PLAYER?**
      3. **AVERAGE VALUE OF SOMETHING?**
      4. **HOW MANY TIMES?**
      5. **TOTAL LIST NUMBER?**
   4. **Close the File**