1. Nick Wright CSC406 Problem 1
2. Java Code

/\* This program was created to simulate a computer system running with the use of threads.  
\* The program contains the classes problem1, GUI, WP, DataStorage, and PrinterDevice.\*/  
  
import java.io.\*;  
import java.io.File;  
import java.lang.\*; //allows for threads to be created as objects from Thread class  
  
/\* public class problem1 contains the main method \*/  
public class problem1 {  
 public static void main(String[] args) throws Exception{  
 //creating a printwriter  
 PrintWriter outf1;  
 outf1=new PrintWriter(new File("problem1output.txt"));  
  
 //creating the Runnables  
 Runnable GUI300 = new GUI(300, outf1);  
 Runnable WP1000 = new WP('A',50\*10\*2,outf1);  
 Runnable DS2500 = new DataStorage(2500,outf1);  
 Runnable Print3600 = new PrinterDevice(3600,outf1);  
  
 //creating the threads  
 Thread thread1 = new Thread(GUI300);  
 Thread thread2 = new Thread(WP1000);  
 Thread thread3 = new Thread(DS2500);  
 Thread thread4 = new Thread(Print3600);  
  
 //setting priorities for the threads  
 thread1.setPriority(Thread.*MIN\_PRIORITY*);  
 thread2.setPriority(Thread.*MAX\_PRIORITY*);  
 thread3.setPriority(Thread.*NORM\_PRIORITY*);  
 thread4.setPriority(Thread.*NORM\_PRIORITY*);  
  
 //starting all threads  
 thread1.start();  
 thread2.start();  
 thread3.start();  
 thread4.start();  
  
 outf1.flush();  
 }  
}  
  
/\* class GUI implements Runnable and contains a constructor for GUI as well as a run method. \*/  
class GUI implements Runnable{  
 private int polls; //the number of polls we will have (polls = seconds\*2)  
 private PrintWriter outf; //this is the output text file from this thread  
  
 public GUI(int seconds, PrintWriter out1){  
 polls=seconds\*2;  
 outf=out1;  
 }//end of GUI constructor  
  
 public void run(){  
 for(int ngp=1; ngp<=polls; ngp++){  
 System.*out*.println("GUI poll " + ngp);  
 //creating a try catch block to use Thread.sleep  
 try {  
 Thread.*sleep*(2);  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 //yielding the thread after each poll  
 Thread.*yield*();  
 }  
 }//end of run  
}//end of GUI  
  
/\* class WP implements Runnable and contains a constructor for WP as well as a run method. \*/  
class WP implements Runnable{  
 private int NumtoPcs; //this is the number of Characters to process  
 private PrintWriter outf; //this is the output text file from this thread  
 private char c; //this is character to process  
 private int numgps; //this is the number of 10 character groups to process  
 private int firstchar; //this is the number of the beginning character on this process group  
 private int lastchar; //this is the number of the last char in this process group  
  
 public WP(char ctoPcs, int num, PrintWriter out1){  
 NumtoPcs=num;  
 outf=out1;  
 c=ctoPcs;  
 //calculate how many 10 character groups to process  
 numgps=NumtoPcs/10;  
 if((NumtoPcs%10)!=0){  
 numgps++;  
 }//if statement adds an extra group if the number of characters is not a multiple of ten  
 }//end of constructor  
  
 //creation of the run method  
 public void run(){  
 for(int ngp=1; ngp<=numgps; ngp++){  
 //this is the group loop for the WP. The WP process NumtoPcs in groups of 10 char at a time  
 firstchar=(ngp-1)\*10+1; //this is the first character to be processed in this group  
 lastchar=firstchar+9; //this is the last character to be processed in this group  
 if((ngp==numgps)&&((NumtoPcs%10)!=0)){  
 lastchar=NumtoPcs;  
 }  
 for(int icr=firstchar; icr<=lastchar; icr++){  
 System.*out*.print(" WP"+icr+" ");  
 }  
 System.*out*.println();  
 //System.out.println("Yielding thread");  
 Thread.*yield*();  
 }//end of this group  
 outf.flush();  
 }//end of run  
}//end of class WP  
  
/\* class DataStorage implements Runnable and contains a constructor for DataStorage as well as a run method. \*/  
class DataStorage implements Runnable{  
 private int NumtoPcs; //this is the number of characters to process  
 private PrintWriter outf; //this is the output text file from this thread  
 private int numgps; //this is the number of groups to process  
 private int firstchar; //this is the first character in the group  
 private int lastchar; //this is the last character in the group  
  
 public DataStorage(int num, PrintWriter out1){  
 NumtoPcs = num;  
 outf=out1;  
 numgps=NumtoPcs/20; //the number of groups to process must be divided by 20  
 if((NumtoPcs%20)!=0){  
 numgps++;  
 }//if statement adds an extra group if the number of characters is not a multiple of twenty  
 }//end of DataStorage constructor  
  
 public void run(){  
 for(int ngp=1; ngp<=numgps; ngp++){  
 firstchar=(ngp-1)\*20+1; //this is the first character to be processed in this group  
 lastchar=firstchar+19; //this is the last character to be processed in this group  
 if((ngp==numgps)&&(NumtoPcs%20)!=0){  
 lastchar = NumtoPcs;  
 }  
 for(int icr=firstchar; icr<=lastchar; icr++){  
 System.*out*.print(" DS"+icr+" ");  
 }  
 System.*out*.println();  
 try {  
 Thread.*sleep*(1);  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 if(ngp%3==0||ngp==numgps){//adding this if condition so that the thread is yielded after 3 cycles  
 Thread.*yield*();  
 }  
 }  
 outf.flush();  
 }//end of run  
}//end of DataStorage  
  
/\* class PrinterDevice implements Runnable and contains a constructor for PrinterDevice as well as a run method. \*/  
class PrinterDevice implements Runnable{  
 private int NumtoPcs; //this is the number of characters to process  
 private PrintWriter outf; //this is the output text file from this thread  
 private int numgps; //this is the number of groups to process  
 private int firstchar; //this is the first character in the group  
 private int lastchar; //this is the last character in the group  
  
 public PrinterDevice(int num, PrintWriter out1){  
 NumtoPcs = num;  
 outf=out1;  
 numgps = NumtoPcs/60; //number of characters to process divided by 60 in order to get number of groups  
 if((NumtoPcs%60)!=0){  
 numgps++;  
 }  
 }  
  
 public void run(){  
 int lineNum=1; //an integer to hold the line number  
 for(int ngp=1; ngp<=numgps; ngp++){  
 firstchar=(ngp-1)\*60+1; //this is the first character to be processed in this group  
 lastchar=firstchar+59; //this is the last character to be processed in this group  
 if((ngp==numgps)&&(NumtoPcs%60)!=0){  
 lastchar = NumtoPcs;  
 }  
 System.*out*.println("Print Line " + lineNum + " characters " + firstchar + " thru " + lastchar);  
 System.*out*.println();  
 lineNum++; //line number increases after each loop  
 if(ngp%6==0||ngp==numgps){//adding this if condition so that the thread is yielded after 3 cycles  
 //System.out.println("Yielding thread");  
 Thread.*yield*();  
 }  
 }  
 outf.flush();  
 }//end of run  
}//end of PrinterDevice

1. Class Documentation

public **class problem1**

**problem1 Data**

The following data is within the main method of this class:

PrintWriter outf1;

Runnable GUI300;

Runnable WP1000;

Runnable DS2500;

Runnable Print3600;

Thread thread1;

Thread thread2

Thread thread3

Thread thread4

**problem1 Functions**

public static void main(String[] args) throws Exception

**class GUI** implements Runnable

**GUI Data**

private int polls;  
private PrintWriter outf;

**GUI Functions**

public GUI(int seconds, PrintWriter out1)

public void run()

**class WP** implements Runnable

**WP Data**

private int NumtoPcs;

private int PrintWriter outf;

private char c;

private int numgps;

private int firstchar;

private int lastchar;

**WP Functions**

public WP(char ctoPcs, int num, PrintWriter out1)

public void run()

**class DataStorage** implements Runnable

**DataStorage Data**

private int NumtoPcs

private PrintWriter outf;

private int numgps;

private int firstchar;

private in lastchar;

**DataStorage Functions**

public DataStorage(int num, PrintWriter out1)

public void run()

**class PrinterDevice** implements runnable

**printerDevice Data**

private int NumtoPcs

privtae PrintWriter outf;

private int numgps;

private int firstchar;

private int lastchar;

**PrinterDevice Functions**

public PrinterDevice(int num, PrintWriter out1)

public void run()