**《Java程序设计》上机作业8**

|  |  |
| --- | --- |
| **学号：** | **202103151422** |
| **姓名：** | **温家伟** |
| **班级：** | **大数据分析2101** |
| **日期：** | **2024.1.10** |

1. 用继承Thread类和实现Runnable接口等两种方式分别编写一个线程程序PrintLetter.java，输出从a到z的26个字母，要求每隔1～2秒钟输出一个字母。

|  |
| --- |
| * **问题分析与解答**   在线程里输出a到 z 26个字母，和书上的卖票代码很像，每隔1~2秒钟加上线程睡眠的代码即可。 |
| * **代码实现**   **继承Thread类**  package test11;  public class PrintLetter extends Thread {  private char Char\_a = 'a';  public void run() {  for (int i = 0; i < 26; i++) {  System.out.print(this.Char\_a);  System.out.print(' ');  this.Char\_a++;  try {  sleep(2000);  }  catch (InterruptedException e) {  System.out.println("线程被唤醒");  }  }  }  public static void main(String[] args) {  PrintLetter Pt = new PrintLetter();  Pt.start();  }  }  **实现Runnable接口：**  **package test11;**  **import java.lang.Thread;**  **public class PrintLetter implements Runnable {**  **private char Char\_a = 'a';**  **public void run() {**  **for (int i = 0; i < 26; i++) {**  **System.out.print(this.Char\_a);**  **System.out.print(' ');**  **this.Char\_a++;**  **try {**  **Thread.sleep(1000);**  **}**  **catch (InterruptedException e) {**  **System.out.println("线程被唤醒");**  **}**  **}**  **}**  **public static void main(String[] args) {**  **PrintLetter Pt = new PrintLetter();**  **Thread thread = new Thread(Pt);**  **thread.start();**  **}**  **}** |
| * **运行结果**   **运行结果都是这个：** |
| * **思考及总结**   这道题要注意线程睡眠要加异常判断。 |

1. 利用多线程编写一个求解某范围素数的程序FindPrime.java,每个线程负责1000范围：线程1找1-1000；线程2找1001-2000；线程3找2001-3000。请编写程序将每个线程找到的素数打印出来。

|  |
| --- |
| * **问题分析与解答**   创建多线程打印素数，把start和end设置静态方便修改。 |
| * **代码实现**   package test11;  import java.math.\*;  import java.util.ArrayList;  import java.util.List;  public class FindPrime extends Thread{  private static int start = 1;  private static int end = 1000;  public boolean IsPrime(int num) {  if (num == 1 || num == 2) {  return false;  }  for (int i = 2; i <= Math.sqrt(num); i++) {  if (num % i == 0) {  return false;  }  }  return true;  }  public void PrintPrimeFromAToB(int a, int b) {  System.out.println(start + "到" + end +"之间素数如下：");  for (int i = a; i <= b; i++) {  if (IsPrime(i)) {  System.out.print(i + " ");  }  }  System.out.println();  }  public void run() {  PrintPrimeFromAToB(this.start, this.end);  this.start += 1000;  this.end += 1000;  }  public static void main(String[] args) {  FindPrime FP1 = new FindPrime();  FindPrime FP2 = new FindPrime();  FindPrime FP3 = new FindPrime();  List<FindPrime> threads = new ArrayList<>();  threads.add(FP1);  threads.add(FP2);  threads.add(FP3);  for (FindPrime thread : threads) {  try {  thread.start();  thread.join();  } catch (InterruptedException e) {  e.printStackTrace();  }  }  }  } |
| * **运行结果** |
| * **思考及总结**   这道题用到了线程等待join。 |

1. 编写多线程程序TestThread.java模拟两个线程交替执行任务：线程1输出1，线程2输出A，接下来线程1输出2，线程2输出B，如此交替执行下去，直到线程2输出Z完毕。

|  |
| --- |
| * **问题分析**   在上述代码中，我们创建了一个名为PrintTask的任务类，该类实现了Runnable接口。在PrintTask中，我们使用了一个静态变量lock作为线程间的同步对象。同时，还定义了一个静态变量number用于存储数字的输出值，以及一个静态变量letter用于存储字母的输出值。 |
| * **代码实现**   **package test11;**  **class PrintTask implements Runnable {**  **private static final Object lock = new Object();**  **private static int number = 1;**  **private static char letter = 'A';**  **private final boolean isNumberThread;**  **public PrintTask(boolean isNumberThread) {**  **this.isNumberThread = isNumberThread;**  **}**  **public void run() {**  **if (isNumberThread) {**  **printNumbers();**  **} else {**  **printLetters();**  **}**  **}**  **private void printNumbers() {**  **synchronized (lock) {**  **while (number <= 26) {**  **System.out.print(number);**  **number++;**  **lock.notifyAll();**  **try {**  **if (number <= 26) {**  **lock.wait();**  **}**  **} catch (InterruptedException e) {**  **e.printStackTrace();**  **}**  **}**  **lock.notifyAll();**  **}**  **}**  **private void printLetters() {**  **synchronized (lock) {**  **while (letter <= 'Z') {**  **System.out.print(letter);**  **letter++;**  **lock.notifyAll();**  **try {**  **if (letter <= 'Z') {**  **lock.wait();**  **}**  **} catch (InterruptedException e) {**  **e.printStackTrace();**  **}**  **}**  **lock.notifyAll();**  **}**  **}**  **}**  **public class TestThread {**  **public static void main(String[] args) {**  **Thread numberThread = new Thread(new PrintTask(true));**  **Thread letterThread = new Thread(new PrintTask(false));**  **numberThread.start();**  **letterThread.start();**  **}**  **}** |
| * **运行结果** |
| * **思考及总结**   多线程编程中需要注意线程安全性和正确的同步机制，以避免出现竞态条件和死锁等问题。 |

1. 利用多线程同步实现三个窗口卖100张票，每一行座位顺序为“A、B、C、D、E、F”。出票顺序是输出窗口\*售出1A号票、窗口\*售出1B号票等。

|  |
| --- |
| * **问题分析**   需要做线程同步使三个窗口卖票，我使用同步控制块的方式。 |
| * **代码实现**   **package test11;**  **public class tickets implements Runnable{**  **private int ticket = 100;**  **private int line = 1;**  **private char seat = 'A';**  **public void run() {**  **while (true) {**  **synchronized (this) {**  **if (ticket > 0) {**  **System.out.println(Thread.currentThread().getName() + "售出" + line + seat + "号票");**  **ticket--;**  **if (seat == 'F') {**  **seat = 'A';**  **line++;**  **} else {**  **seat++;**  **}**  **} else {**  **break;**  **}**  **}**  **}**  **}**  **public static void main(String[] args) {**  **tickets ru = new tickets();**  **Thread t = new Thread(ru);**  **t.setName("窗口 1");**  **Thread t1 = new Thread(ru);**  **t1.setName("窗口 2");**  **Thread t2 = new Thread(ru);**  **t2.setName("窗口 3");**  **t.start();**  **t1.start();**  **t2.start();**  **}**  **}** |
| * **运行结果**   **窗口1切换到窗口3再切换到窗口2** |
| * **思考及总结**   这道题我学会了使用同步控制块的方式实现线程同步。 |