プログラミングA 第9回・演習　回答

1　演習 1

参考資料：FibonacciThread.java

```
public class FibonacciThread extends Thread {

   private int value = 0;

   public FibonacciThread(int v) {
      value = v;
   }

   public void run() {
      printFibonacci(value);
   }

   public static long fibonacci(int n) {
      return (n == 0 || n == 1) ? 1 : fibonacci(n - 1) + fibonacci(n - 2);
   }

   public static void printFibonacci(int n) {
      System.out.println(Thread.currentThread() + "" + fibonacci(n));
   }

   public static void main(String[] args) {
      long previousTime = 0;

      previousTime = System.currentTimeMillis();
      // 追加された部分
      Thread[] threads = new Thread[10];
      for(int i = 30; i < 40; i++) {
         threads[i - 30] = new FibonacciThread(i);
         threads[i - 30].start();
      }
      try {
         for(int i = 30; i < 40; i++) {
            threads[i - 30].join();
         }
      } catch (InterruptedException e) {
      }
      // ここまで
      System.out.println("Time spent for multi-threaded execution: "
            + (System.currentTimeMillis() - previousTime));

      previousTime = System.currentTimeMillis();
```

```
    for(int i = 30; i < 40; i++) {
      printFibonacci(i);
    }
    System.out.println("Time spent for single-threaded execution: "
        + (System.currentTimeMillis() - previousTime));
  }
}
```

2 演習 2

参考資料：MessagePrint.java

```
class MessagePrint {
  Thread previousThread = null;
  public synchronized void message(String text) throws InterruptedException{
    while(previousThread == Thread.currentThread()) {
      wait();
    }
    previousThread = Thread.currentThread();
    System.out.println(text);
    notifyAll();
  }
}
```

参考資料：ThreadXX2.java

```
class ThreadXX2 extends Thread {
  MessagePrint m = null;
  public ThreadXX2(MessagePrint nm) {
    super();
    m = nm;
  }
  public void run() {
    try {
      while(true) {
        m.message("XX");
      }
    } catch(InterruptedException ie) {
    }
  }
}
```

参考資料：ThreadYY2.java

```
class ThreadYY2 extends Thread {
  MessagePrint m = null;
  public ThreadYY2(MessagePrint nm) {
```

```
      super();
      m = nm;
   }
   public void run() {
      try {
         while(true) {
            m.message("YY");
         }
      } catch(InterruptedException ie) {
      }
   }
}
```

3 演習 3

参考資料：Queue.java

```
public class Queue {
   // ...
   public synchronized void enqueue(int data) throws
   InterruptedException {
      while(isFull()) wait();
      values[last] = data;
      last = (last + 1) % values.length;
      System.out.println("Enqueue: " + data);
      notifyAll();
   }
   public synchronized int dequeue() throws InterruptedException {
      while(isEmpty()) wait();
      int data = values[first];
      first = (first + 1) % values.length;
      System.out.println("Dequeue: " + data);
      notifyAll();
      return data;
   }
   public boolean isFull() {
      return (((last + 1) % values.length) == first);
   }
   boolean isEmpty() {
      return (last == first);
   }
}
```

参考資料：QueueConsumer.java

```
class QueueConsumer extends Thread {
   public void run() {
```

```
    try {
      while (true) {
        int x = queue.dequeue();
        System.out.println(getName()+"  “ +x+" を消費 ");
        sleepRandomly();
      }
    } catch (InterruptedException e) {
    }
  }
  // ...
```

参考資料：QueueProducer.java

```
class QueueProducer extends Thread {
  public void run() {
    try {
      for (int i = 0; i < 30; i++) {
        queue.enqueue(i);
        System.out.println(getName()+"  “ +i+" 追加 ");
        sleepRandomly();
      }
    } catch (InterruptedException e) {
    }
  }
  // ...
```

4　演習 4

参考資料：Room.java

```
import java.util.Random;
public class Room {
  private Random random = new Random();
  private boolean resting = false;
  public void rest() {
    synchronized (this) {
      if (resting)
        return;
      resting = true;
    }
    System.out.println("Start resting :"
        + Thread.currentThread().getName());
    try {
      Thread.sleep(random.nextInt(5000));
    } catch (InterruptedException e) {
    }
    System.out.println("End resting :"
```

```
            + Thread.currentThread().getName());
        resting = false;
    }
}
```

5 演習 5

(5-1)
- ThreadA が X のインスタンスのロックを獲得したままで Y のインスタンスのロクを獲得しようとするのに対して、ThreadB は逆に Y, X の順にインスタンスロックを獲得しようとする。
- そこで、ThreadA が X のロックを獲得した状態で、同時に ThreadB が Y のロッを獲得すると、それ以上、両スレッドとも進行できなくなる (デッドロック)

(5-2) 参考資料：ThreadB.java

```
class ThreadB extends Thread{
    // ...
    public void run(){
        for(int i = 0; i < 10; i++) {
            try {
                synchronized(x) {
                    x.doX(i);
                    synchronized(y) {
                        y.doY(i);
                    }
                }
            } catch(InterruptedException ie) { }
        }
    }
}
```