MULTITHREADING

1.
```
class Counter {
        int count=0;
        int increment() {
              int n=count;
              count = n+1;
              return n;
        }
   }
```
How do you make increment() thread-safe in the sample code above?

a. Implement the java.lang.SingleThreaded interface.
b. Modify increment() to stop all other threads.
c. Declare increment() synchronized.
d. Declare increment() as static final.
e. Declare Counter synchronized

2.
```
Class Sample {
   private final Object build1Lock = new Object ();
   private final Object build2Lock = new Object ();
        public synchronized build () {
              Synchronized (build1Lock) {
                    // … Do Something
              }
        }
   }
```
How do you make the sample code above operate?

a. Put both objects in their own class
b. Put both methods into one class
c. Change “synchronized” to void in both methods.
d. Change “synchronized” to “String” in the second method.
e. Change “synchronized” to “String” in the first method.

3.
```
public class SampleLong {
   //????///
   public void setSample(long sample) {
        this.sample.set(sample);
   }
   public long getSample() {
        return sample.get();
   }
   }
```
Based on the above sample code, which line of code do you insert in the place of ???? for the code to function properly?

a. private atomic long sample = new atomic long();
b. private long sample = new long();
c. private LongAtomic sample = new LongAtomic();
d. private Atomic sample = new Atomic();
e. private AtomicLong sample = new AtomicLong();

4. What is the purpose of acquiring an object's monitor by using a synchronized(object) {} construct before accessing the object's resources?
   a. To prevent multiple threads from accessing the same resource at the same time, which could leave the resource in an undefined state
   b. To prevent a deadlock or race situation from occurring between multiple threads during event handling
   c. To prevent flicker on the user's screen during updates that cover more than 80 percent of the display area
   d. To ensure objects accessed on remote machines have their dates expressed in the local time zone
   e. To establish scheduling priority over other threads so the bytecode interpreter gives more processor time to the synchronized thread

5. Sample Code

```
void shifter(int[] array, int arrayLength) {
      System.out.println("Go");
      synchronized (array) {
      for(int i=1;i<arrayLength;i++) {
            array[i-1] = array[i];
      }
      }
}
```

What does the synchronized array code block do in the sample code above?
   a. It ensures any threads created in the code block begin execution in the array object.
   b. It forces the thread to obtain the monitor for array before continuing.
   c. It sets all methods of the object to synchronized while the code block executes.
   d. It performs the same as declaring shifter() as synchronized.
   e. It forces the virtual machine to treat the array as a static object.

6. public synchronized int doIt(int z) {
int x=5;
while(otherMethod()) {
try {
wait();
} catch(InterruptedException e) {
}
}
for (int i=0;i<z;i++)
count += x;
int result = count - x;
notifyAll();
return result;
}
Based on the sample code above, what is the result of invoking notifyAll() in method doIt()?

a. It informs other threads that this thread has completed the method.
b. It calls the notifyAll() method as defined in this class and does whatever is defined there.
c. It calls the notifyAll() method as overridden by this class's subclass and does whatever is defined there.
d. It allows other threads to enter any synchronized method of this object.
e. It changes all class methods to synchronized until release() is invoked.

7. Scenario
There are a few tasks to run in parallel in Java. Each task returns either success or failure. Each task has an associated deadline. If a task does not finish by the deadline, it is interrupted and returns failure.
If one of the tasks fails, we interrupt all other tasks which are still running.
After all tasks finish, you want the code to return either success if all tasks return success, or failure if at least one task returns failure.
Based on the scenario above, how do you implement this?

a. ParallelService
b. FinishService
c. ImplementService
d. ExecutorCompletionService
e. AnalyserService

8. Sample Code

```
class MyThread extends Thread {
   MyThread() {
     System.out.print(" Sample");
   }
   public void run() {
     System.out.print(" example");
   }
   public void run(String s) {
      System.out.print(" model");
   }
}
class TestThreads {
  public static void main (String [] args) {
    Thread t = new MyThread() {
      public void run() {
        System.out.print(" case");
      }
    };
    t.start();
} }
```

What is the printed when the sample code above is executed?

a. Sample case
b. model
c. example
d. example case
e. Sample

9.

```
public class ThreadUnsafe{
      public static int commonCounter = 0;

      public static int getCount() {return commonCounter;}
      public void addCount(int val){commonCounter += val;}
      public void subtractCount(int val){commonCounter -= val;}
}
```

How do you make the class defined in the sample code above thread-safe?

a. Have ThreadUnsafe implement the java.servlet.SingleThreadModel interface.
b. Store ThreadUnsafe objects in a collection created using Collections.synchronizedCollection().
c. Change the definition of commonCounter to protected static int.
d. Wrap the ThreadUnsafe object within a class that declares all methods are synchronized.
e. Modify getCount(), addCount() and subtractCount() to synchronize on a static object instance.

10.
```
class Receiver extends Thread{
      private InputStream in;
      private Receiver(InputStream in) {this.in=in;}
      public void run() {// … Do some work with the input}
}
class Sender extends Thread {
      private OutputStream out;
      private Receiver(OutputStream out) {this.out=out;}
      public void run() {// … Generate the output and sent it
                          //…to the output stream. }
}
Void main(String args[]) throws IOException {
      InputStream in; OutputStream out;
      // ????
      new Receiver(in).start();
      new Sender(in).start();
}
```
Which line of code do you insert in place of ???? in the sample code above to connect the Receiver thread to the Sender thread?

a. out = new FileOutputStream("l");in = new FileInputStream("l");
b. in = new PipedInputStream(); out = in.getOutputStream();
c. PipedStream.getStreamPair(in,out);
d. in = new PipedInputStream(); out = new PipedOutputStream((PipedInputStream)in);
e. PipedStream p= new PipedStream;in=p.in;out=p.out;

11.
```
public class Outer{
    private static class Inner Inner implements Runnable{
      private int a;
      Inner(){a=1;}
      //some code
    }
    public static Runnable createInner(){
            return new Inner();
    }
}
```
Given the class Outer defined in the sample code above, which one of the following code fragments obtains a reference to a new Inner Object from another class

a. Outer.createInner();
b. new Outer.new Inner();
c. Outer.new Inner();
d. new Outer.Inner();
e. Outer.Inner.new();

12.
```
// Creates a thread pool and separate tasks to count
 // the length of each word passed in on the command line.
import java.util.*;
import java.util.concurrent.*;
public class MyExample {
      public static class MyWordLength implements XXX {
            private String word;
            public MyWordLength(String word) {
                  this.word = word;
            }
            public Integer call() {
                 return Integer.valueOf(word.length());
            }
      }
      public static void main(String args[]) throws Exception {
           ExecutorService pool = Executors.newFixedThreadPool(3);
           Set<Future<Integer>> set = new HashSet<Future<Integer>>();
           for (String word: args) {
                XXX<Integer> xxx = new MyWordLength(word);
                Future<Integer> future = pool.submit(xxx);
                set.add(future);
           }
           int sum = 0;
           for (Future<Integer> future : set) {
                  sum += future.get();
            }
            System.out.printf("The sum of lengths is %s%n", sum);
            System.exit(sum);
       }
}
```

In the sample code above, which interface name do you insert in place of XXX for each task that needs to execute to create a thread pool of size 3?

a. Callable
b. Future
c. Executor
d. Doable
e. Accessible