Solutions to Quiz 3 (April 11, 2014)

Problem 1 (Deadlock) (20 points).
In the skeleton code below three threads (1, 2, 3) are trying to acquire three locks (A, B, C). Suppose the code deadlocks, and circle the line where each thread is blocked. If a thread has run to completion in your scenario, circle “thread finished.” If the code might deadlock in more than one way, then just select one possible way it could deadlock.

Hint: Try to rule out places the code cannot block first.

Thread 1
synchronized(A) {
    ...
}
synchronized(C) {
    synchronized(A) {
        ...
    }
synchronized(C) {
    }
synchronized(B) {
        ...
    }
synchronized(B) {
    }
thread finished

Thread 2
synchronized(C) {
    synchronized(A) {
        ...
    }
synchronized(A) {
        ...
    }
synchronized(B) {
    }
synchronized(B) {
    }
thread finished

Thread 3
synchronized(C) {
    synchronized(A) {
        ...
    }
synchronized(A) {
    }
synchronized(C) {
    }
synchronized(C) {
    }
thread finished

Solution. Thread 1: thread finished or synchronized(C) or synchronized(B)
Thread 2: synchronized(B)
Thread 3: synchronized(C)
OR
    Thread 1: synchronized(B)
    Thread 2: thread finished or synchronized(C) or synchronized(B)
    Thread 3: synchronized(C)

The next few questions use the following code.

/** A brush. Immutable and safe for concurrency. */
public class Brush { ... }

/** Commission for a painting. Immutable and safe for concurrency. */
public class Commission { ... }
/** A painting palette. Mutable. */
public class Palette {

/** An artist who paints paintings. Designed to be the runnable for a thread. */
public class Painter implements Runnable {

private final Set<Brush> brushes;
private final Palette palette;
private final BlockingQueue<Commission> commissions;

public Painter(Set<Brush> brushes, BlockingQueue<Commission> commissions) {
    this.brushes = brushes;
    this.palette = new Palette();
    this.commissions = commissions;
}

private void pick() {
    synchronized (palette) {
        ... // uses: palette (no threads created)
    }
}

private void paint() {
    synchronized (brushes) {
        ... // uses: Brushes in brushes (no threads created)
    }
}

private void mix() {
    synchronized (this) {
        ... // uses: palette and a Brush from brushes (no threads created)
    }
}

public void run() {
    while (true) {

        // when we want a new commission:
        Commission c = commissions.take();

        ... // use methods above to paint the commissioned painting (no threads created)
    }
}
}
/** A patron who commissions paintings. Designed to be the runnable for a thread. */
public class Patron implements Runnable {
    private final BlockingQueue<Commission> commissions;

    public Patron(BlockingQueue<Commission> commissions) {
        this.commissions = commissions;
    }

    public void run() {
        while (true) {
            // if we have money:
            commissions.add(new Commission(/* ... details ... */));
        }
    }
}

/** A studio where patrons and artists interact. Designed to be the main method of a program. */
public class Studio {
    private static final BlockingQueue<Commission> commissions = new LinkedBlockingQueue<>();
    private static final Set<Brush> brushes = new HashSet<>();

    public static void main(String[] args) {
        for (int i = 0; i < 10; i++) {
            new Thread(new Patron(commissions)).start();
        }

        // add some Brushes to brushes
        for (int i = 0; i < 5; i++) {
            new Thread(new Painter(brushes, commissions)).start();
        }
    }
}

Problem 2 (Concurrency Concepts) (20 points).
Circle all correct answers for the following questions.
(a) Studio.commissions is used to implement:
   A. lock ordering
   B. message passing
   C. immutable requests
   D. a network socket
   E. interruption
   F. poison pills

Solution. B. Credit also given for C, because the commissions queue contains immutable requests.

(b) Painter is a:
Solutions to Quiz 3 (April 11, 2014)

A. producer
B. consumer
C. message
D. runnable
E. type

**Solution.** B, D, E.

(c) **Commission** is a:

A. producer
B. consumer
C. message
D. runnable
E. type

**Solution.** C, E.

(d) **Patron** is a:

A. producer
B. consumer
C. message
D. runnable
E. type

**Solution.** A, D, E.

**Problem 3** (Concurrency Behavior) (20 points).

**Circle all correct answers** for the following questions.

(a) Which of the following is true:

A. constructing a new **Painter** can block
B. in **Painter**.pick, synchronizing on **palette** can block
C. in **Painter**.paint, synchronizing on **brushes** can block
D. in **Painter**.mix, synchronizing on **this** can block
E. in **Painter**.run, calling **commissions**.take() can block

**Solution.** C, E.

(b) Based on their specifications, without writing any locking code, it would be safe for concurrent **Painters** to share:

A. **Studio**.commissions
B. **Studio**.brushes
C. **Brush** objects in **Studio**.brushes
Solutions to Quiz 3 (April 11, 2014)

D. Palette objects in Painters
E. Commission objects

Solution. A, C, E.

(c) From the code as given, Painters do share with other Painters:
A. Studio.commissions
B. Studio.brushes
C. Brush objects in Studio.brushes
D. Palette objects in Painters
E. Commission objects

Solution. A, B, C.

(d) While this system is running:
A. There will never be more than 10 Commission objects
B. There will never be more than 10 Commission objects in commissions
C. There will never be two Painters in the synchronized block of pick simultaneously
D. There will never be two Painters in the synchronized block of paint simultaneously
E. There will never be two Painters in the synchronized block of mix simultaneously
F. There will never be two Painters working on the same Commission

Solution. D, F.

(e) When will the system stop running?
A. When one Patron adds a poison pill to commissions
B. When every Patron adds a poison pill to commissions
C. When one Painter finishes a Commission
D. When every Painter finishes a Commission
E. When all Commissions are finished
F. Never

Solution. F.

Problem 4 (Thread Safety Arguments) (20 points).
For each of the objects below, if there is a thread safety bug, say what the bug is in one sentence. If there is not a thread safety bug, select the most appropriate category and make a one-sentence thread safety argument. Choose only one answer for these questions. If you feel that more than one choice applies, choose the best one and give a clear one-sentence explanation.

(a) Studio.commissions
A. Thread safety bug: ____________________________________________________
B. Confinement: _____________________________________________________
C. Immutability: ______________________________________________________
D. Threadsafe types: ___________________________________________________
E. Synchronization: _____________________________________________________

Solution. D. BlockingQueue is threadsafe.

Studio.brushes

A. Thread safety bug: _____________________________________________________
B. Confinement: _________________________________________________________
C. Immutability: _________________________________________________________
D. Threadsafe types: _____________________________________________________
E. Synchronization: _____________________________________________________

Solution. A. brushes is a HashSet, which is not a threadsafe type; it’s not confined; it isn’t protected by synchronization, e.g. mix() uses brushes without synchronizing on it.

Brush objects in Studio.brushes

A. Thread safety bug: _____________________________________________________
B. Confinement: _________________________________________________________
C. Immutability: _________________________________________________________
D. Threadsafe types: _____________________________________________________
E. Synchronization: _____________________________________________________

Solution. Either C or D. Brush is immutable and threadsafe.

Palette objects in Painters

A. Thread safety bug: _____________________________________________________
B. Confinement: _________________________________________________________
C. Immutability: _________________________________________________________
D. Threadsafe types: _____________________________________________________
E. Synchronization: _____________________________________________________

Solution. B. Each Palette is only accessible from one Painter thread.

Commission objects

A. Thread safety bug: _____________________________________________________
B. Confinement: _________________________________________________________
C. Immutability: _________________________________________________________
D. Threadsafe types: _____________________________________________________
E. Synchronization: _____________________________________________________
Solution. Either C or D. Commission is immutable and thread-safe. An argument can also be made for B, since a Commission is first confined to the Patron who creates it, who then gives up the reference entirely to the Commissions queue, and then the reference becomes confined to the Painter who takes it from the queue.

Problem 5 (Grammars) (20 points).
Consider this grammar:

\[
S ::= K \times C^+
\]
\[
X ::= B \mid X B
\]
\[
C ::= A \mid D
\]

(a) List the nonterminals of this grammar. (Warning: do not rely on capitalization to tell you whether something is a nonterminal.)

Solution. S, X, C

(b) Suppose S is the starting nonterminal. Which of the following are valid sentences in the language specified by the grammar? Circle the ones that are valid. (Whitespace is unimportant.)

1. K
2. K B B Y
3. K B D
4. B D
5. K B A D A D D

Solution. Valid sentences are 3 and 5.