

SYSC 3303 Summer 2014 Quiz #1

Name:

Student Number:

Q1(/11)	Q2(/6)	Q3(/8)	Q4(/15)	Total(/40)

Question 1 (1+1+2+1+2+2+2=11):

- a) Give the name of a hard real-time system.

Example: Anti-Lock Braking System (ABS) of a car

- b) Briefly describe (one or two sentences) your hard real-time system given in part a).

The ABS on a car is a system that will prevent the brakes from locking when the driver presses hard on the brake pedal.

- c) What is the difference between a soft real-time system and a hard real-time system?

In a hard real-time the deadlines are critical, missing them leads to death and destruction. In a soft real-time system, the deadlines are important but may occasionally be missed and the system will continue to function.

- d) Why is your example a hard real-time system?

If the ABS does not meet the deadline after the brake pedal is pressed, a car accident could result.

- e) What are the stimuli of your system?

The inputs are presses on the brake pedal, as well as feedback from the sensors on the wheels of the car.

- f) Give an example of a fault (in a component or network) that your system is able to deal with, and explain how it deals with it.

Even if a sensor on one of the four wheels is not working, the ABS system will still continue to work. It will use the input from the remaining sensors. In this circumstance, a warning light should inform the driver of an issue with the sensor.

- g) List two or three states of the environment that your system is able to deal with.

The ABS system will work properly regardless of the weather, e.g. sunshine, darkness, heat, cold, rain, snow, etc., and regardless of the road surface, e.g. asphalt, gravel, dry, wet, icy, potholes, etc.

Question 2 (2+2+2=6):

- a) What are the two ways that we can create a new thread in a Java program?

Subclassing Thread, and implementing the Runnable interface.

- b) For each case, write the first line of the class (up to the "{"):

class Example1 extends Thread { ...

class Example2 implements Runnable { ...

- c) Why isn't the first way that we discussed in class sufficient? (Why are there two ways?)

As Java has only single inheritance, the first way (subclassing Thread) would not suffice if we wanted to subclass a different class.

Question 3 (1+1+2+2+2=8):

- a) What layer of our five level protocol stack does TCP implement?

Transport

- b) What layer of our five level protocol stack does UDP implement?

Transport

- c) What is the key difference between TCP and UDP?

TCP establishes a virtual connection and is reliable. UDP is connectionless and unreliable.

- d) What is the key difference between UDP and IP?

IP identifies a machine (computer). UDP identifies a port (associated with an application) on the machine.

- e) For a client to send a request to a server, it needs two pieces of information:

The __IP address__ of the server host, and the __well-known__ port.

Question 4 (3 each = 15, plus bonus marks if you find more than five): // Corrected May 21st

Identify at least five mistakes in the code below. For each mistake, indicate what is wrong, why it is wrong, and how to correct it. All the mistakes relate to concepts introduced in this course.

```
class Box extends Thread {
    private Object contents = null; // box contents
    private boolean empty = true; // is box empty?
    // put method omitted (this is not a mistake!)
    public Object get () { // the get method
        if (empty) {
            sleep();
        }
        Object item = contents;
        contents = null;
        empty = true;
        notify();
        return item;
    }
}
```

Mistake #0:

Box is a public class.

This doesn't really relate to the course material, and public is the default, but I will give you this one!

First line should be: public class Box ...

Mistake #1:

Box extends Thread.

Box is a passive, not an active class, so it is not a Thread.

First line should be: public class Box {

Mistake #2:

The get method is not synchronized.

For the Box method to work properly in all circumstances, all its methods must be critical sections.

First line of get should be: public synchronized Object get() {

[Note: synchronized may be elsewhere in the first line, but must appear.]

Mistake #3:

sleep() needs a parameter.

sleep() will not compile without a parameter.

Should be: sleep(x); // where x is a number of milliseconds

Mistake #4:

sleep() is a static method.

sleep() which is equivalent to this.sleep() will not compile (with or without a parameter).

Should be: Thread.sleep(x);

Mistake #5:

sleep() should not be used here.

We want to give up the lock, not keep the lock, to allow other threads to proceed.

Should be: wait();

Mistake #6:

wait is inside an if, not a while loop.

For safety and liveness, wait must always be invoked inside a while loop.

Should be: while (empty) { wait(); }

Mistake #7:

wait is not inside a try/catch for InterruptedException.

InterruptedException is a caught exception and the program will not compile without this.

Should be: try{ wait(); } catch (InterruptedException e) { return null; }

[Note: students do not have to identify what goes in the final braces for full marks.]

Mistake #8:

notify should be notifyAll.

As producers and consumers are waiting for different conditions (empty and !empty) notify is not suitable.

Should be notifyAll();