Exam 2017 23.11.17

Question 1: Thread programming

1.a.1

public class Event{

semaphore signaledState; %0 1 if unsignaled/signaled

function wait() {

if (not signaledState){

thread.block()

} else {

//return without blocking the caller

}

}

function signal() {

singnaledState.signal() %change variable to 1

}

}

1.b

One solution is to make sure only one student grabs a piece of pizza at a time. If no piece is left, call the pizzeria and go to sleep. If pizzeria is already called go to sleep.

while (true) {

if (pizzaLeft)

pizzaLock.lock();

pick up a piece of pizza;

pizzaStack.pop()

if last pizza slice:

set pizzaLeft = false

pizzaPlace.order()

pizzaLock.unlock();

study while eating the pizza;

} else {

Sleep;

}

PizzaPlace

Student.WakeUp

1.c

Say if some thread have a task of inverting a value. If it is ran twice, the value wil go back to its original state and not the inverted sate as wanted.

OR: if the first instance of the thread acquire a lock and then is put to sleep, then the second instance executes, because it has the same ID it will already have the lock, it will run and then unlock the lock. The first instance will then continue doing its thing, however another thread might have altered the resource is was supposed to use, as the lock was unintentionally unlocked by the second instance of the first thread.

Question 2: Reactive programming

2.a

Reactive programming is programming using asynchronous data streams.

It is about programming with messages. You register your interest in some data and then receive a callback when it is available.

2.b

An observable is something that emits values. An observer can subscribe to an observable, in order to receive these emitted values.

An iterable is something with an iterator.

The observable is a producer who will push values to the consumer.

The iterable is a consumer who will pull values from the producer. It is synchronous because it will wait for the value to be available. The observable will just push a value whenever it is available and is asynchronous.

2.c

It will take all values below 5 and multiply them with 10, and then print them. The output will thus be:

10 20 30 40

**Question 3: programming for cloud**

**3a**

Hot: real time monitoring. Latency < 1 min. Used for alerting, event counts

Warm: close to real time. Latency < 5 min. Used for text based logs, debug messages, exceptions, performance numbers. Data has short retention (kept for about 2 weeks)

Cold: historical completeness. Long retention time. Text based logs. Metrics and counters. Used for capturing trends over time. Possible to create expensive aggregates. Needs to lack behind for completeness (e.g. due to time zones)

**3.b.1**

A UTF-16 string format is a factor 2 more expensive in size than a bit. A string guid is 32 characters requiring 2 bit per character, thus 64 bits in total. The bit guid is only 16 bit in total, thus it is only ¼ the size of a string format.

**3.b.2**

Pros of using bit representation is the smaller size.

**3.b.3**

Because we have A LOT of data, in such a large scale system. Every bit counts.

**3.c.1**

Data is susceptible to interpretation. Schema is flexible, data is flexible and can be arbitrary. Not performant but flexible.

**3.c.2**

Hard schema use data that can be quantifiable. Fixed data types, fixed columns. Performant but rigid.

**3.c.3**

Hybrid tries to get the best of both worlds. E.g. som field may always exist and always be of a certain type, for example a user ID, this is a hard type. Some field like a stack trace may not exist and can contain different amount and type of data. This should be a soft field.

**3.d**

Scrubbers can be used to anonymize data, but still making it possible to find user data, by using annotations, so that it can be deleted, which must be possible to achieve compliance.

Encryptions can always be turned back using the right key. A hash cannot necessarily be turned back. Hashes are good to use for e.g. passwords, where you dint want to know the real value.

Scrubbing when and where? When the log is saved to disk by the computer who saves it.   
It has to be done as soon as possible.