Final Review Sample Problems

For these problems, assume the compiler will not change the relative order of instructions within a thread, and only worry about the interleaving of code between two threads.

- What is the following function checking? I.e. for what kind of inputs does funcA return true?

  ```
  (define (funcA x)
    (cond
      ((null? x) #f)
      ((integer? x) #t)
      (#t (and (funcA (car x)) (null? (cdr x))))
    )
  )
  ```

- Consider the following code:

  ```
  int data = 0;
  int t = 10;
  boolean flag = false;
  Object m = new Object();
  ```

  ```
  Thread 1                                  Thread 2

  flag = true;
                                            if (flag)
                                                synchronized (m) {
                                                    t = data;
                                                }
  synchronized(m) {
      data = 10;
  }
                                            print(t);
  ```

  Assuming the code executes in the order implied by the spacing, is there a data race? Justify your answer using vector clocks (Assume Thread 1 starts with clock [1,0] and Thread 2 starts with clock [0, 1]).

  Considering again this execution order of the code, does it exhibit atomicity? (i.e. will the state of the program at the end be the same as some serial execution of this code?)

- Consider the following code:

  ```
  int data = 0;
  boolean flag = false;
  Object m = new Object();
  ```

Thread 1

```
flag = true;
synchronize(m) {
    data = 10;
}
```

Thread 2

```
boolean f;
synchronize(m) {
    f = flag;
}
if (f) {
    data = 100;
}
print(data);
```

For this code, there is at least one execution order (interleaving) which has a data race, and at least one execution order which does not have a data race. Find an example of both.