COMP 2710 – Spring 2020
Final Exam (take-home)

What to submit:

1. A .pdf file of your answers with a file name of "**FinalExam_LastName_UserID.pdf**". (Please type answers with your keyboard, hand-written answers are NOT accepted).
    1.1. Create a .doc file.
    1.2. Type answers through a keyboard.
    1.3. Save it as a .pdf file.
    1.4. Submit it on Canvas.

Maximum points possible: 50(.pdf) + 50 (coding part) = 100

Time: 11:59pm CST Apr 29th - 11:59pm CST May 1st

1. Multiple Choice (14 points, 2 points/Question). Hint: Pick one choice for each question.

(1) We create a dynamic array as follows:

Data type: Double pointer   variable name d;
d = new double[10];

Which of the following statement delete the dynamic array?

a) delete d;

b) delete & d;

c) delete * d;

[✓ d) delete [] d;]

(2) Which of the following statement related to pointers is **incorrect**?

a) Pointers are memory addresses of variables

b) Memory addresses are pointers that pointing to variables of a given data type

c) In the call-by-reference approach, the addresses of arguments are passed

[✓ d) None of the above is correct]

(3) Suppose we have the following definitions and assignments:

double *p1, *p2, v;

p1 = &v;

v = 9.9;

p2 = p1;

Which of the following statement is **incorrect**?

a) [✓ *p1 == &v]

b) *p2 == 9.9

c) p2 == &v

d) p1 == p2

(4) Pointer variables are memory addresses and can be assigned to one another without regard to type.

a) True   [✓ b) False]

(5) Recursive functions can be accomplished in one step, namely repeated calls to itself.

a) True   [✓ b) False]

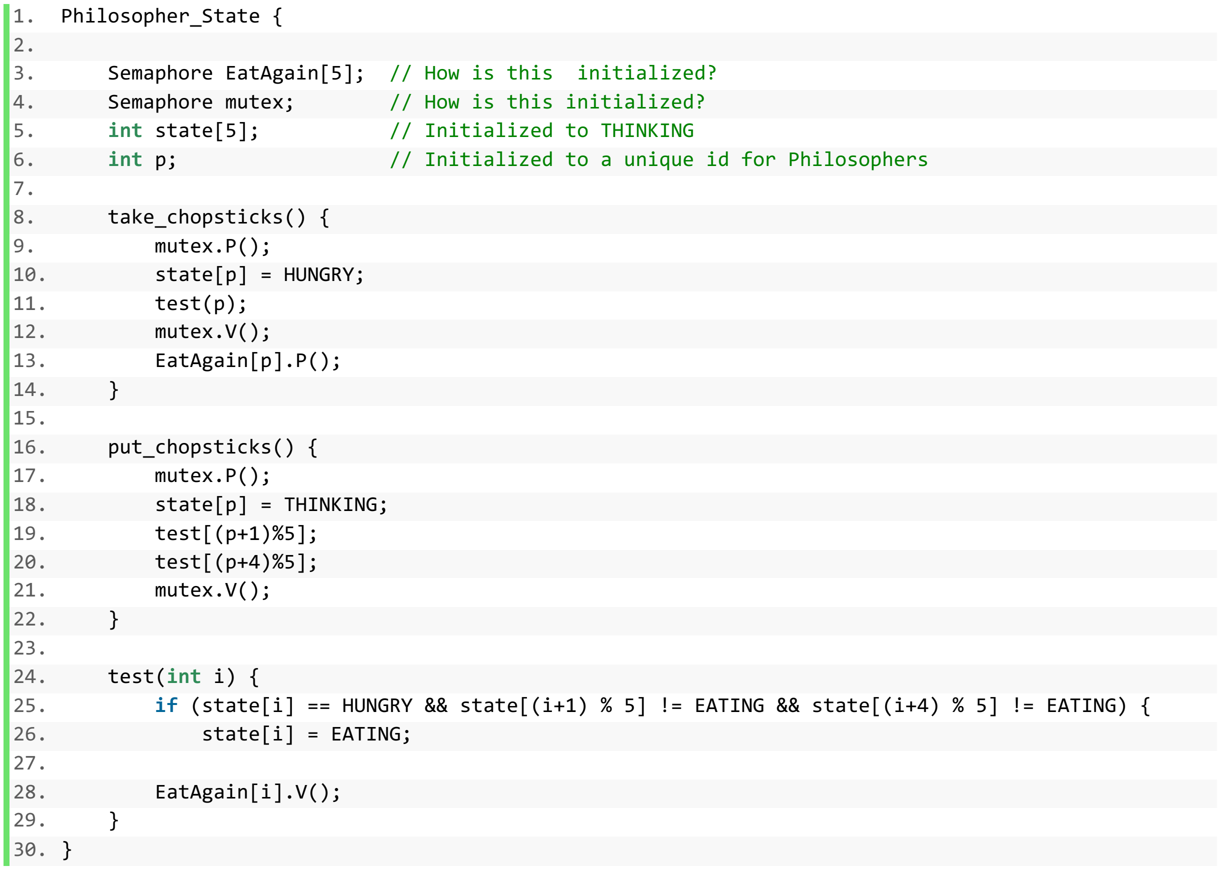
(6) A recursive function with parameter N counts up from any negative number to 0. An appropriate base case would be N == 0.

[✓ a) True]   b) False

(7) A recursive function can have two base cases, such as N == 0 returning 0, and N == 1 returning 1.

[✓ a) True]   b) False

8. Revised solution 2 with three State variables regarding the Dining-Philosopher problem. (36 points)



```
Philosopher_State {

    Semaphore EatAgain[5];  // How is this  initialized?
    Semaphore mutex;        // How is this initialized?
    int state[5];           // Initialized to THINKING
    int p;                  // Initialized to a unique id for Philosophers

    take_chopsticks() {
        mutex.P();
        state[p] = HUNGRY;
        test(p);
        mutex.V();
        EatAgain[p].P();
    }

    put_chopsticks() {
        mutex.P();
        state[p] = THINKING;
        test[(p+1)%5];
        test[(p+4)%5];
        mutex.V();
    }

    test(int i) {
        if (state[i] == HUNGRY && state[(i+1) % 5] != EATING && state[(i+4) % 5] != EATING) {
            state[i] = EATING;

        EatAgain[i].V();
    }
}
```

8.1. Please carefully review Solution 2 to list three states. (3 points)

THINKING, HUNGRY, EATING

8.2. How should the Semaphore elements of EatAgain be initialized? (3 points)

sem_t EatAgain[5];

8.3. How should the Semaphore mutex be initialized? (3 points)

sem_t mutex;

8.4. What is the maximum number of Philosophers that can be waiting on a Semaphore element mayEat[i] at any given time? (3 points)

1

8.5. What is the maximum number of Philosophers that can be waiting on mutex at any given time? (3 points)

4

8.6. Does the code work correctly if the statement EatAgain[i].P() is moved before mutex.V() in take_chopsticks()? Briefly explain it. (4 points)

It wouldn't work because the variable i in EatAgain[i].P() is not defined in the take_chopsticks method and it would be in a critical condition

8.7. Does the code work correctly if the statements test((i+1)%5) and test((i+4)%5) are moved before state[i]=THINKING() in put_chopsticks()? Briefly explain it. (4 points)

It wouldn't work because the program has to go to eating to thinking before it can see if a neighbor is set to eat

9. In the Fig. 0, suppose A and B are making simultaneous transfers between two accounts in a bank. Please predict potential threats for this transaction. (5 points)

Potential threats include mutual exclusion, hold and wait, no pre-emption, and circular wait

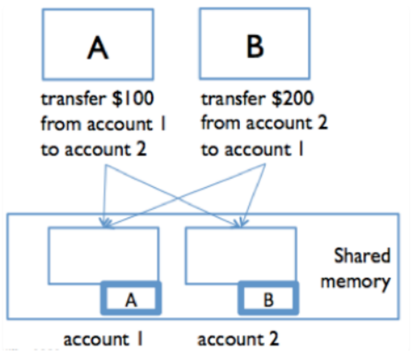

A
B
transfer $100 from account 1 to account 2
transfer $200 from account 2 to account 1
Shared memory
A
B
account 1
account 2

Figure 0

10. Please summarize source of major software developers' headaches from the concurrency mechanism. List at least 4 drawbacks. (8 points)

1) Debugging
2) Testing
3) Managing the concurrency
4) Writing the code