Assignment 3
Due Friday March 15th by 11:59pm
Submit using the following links: Individual problems: https://classroom.github.com/a/NCTFFUMEx
Programming problems: https://classroom.github.com/g/3OxGJwbl

Part 1 – Individual Problems

Problem 1 – 5.2 in 9th edition
Consider Dekker’s algorithm written for an arbitrary number of processes by changing the statement executed when leaving the critical section from:

\[
\text{turn} = 1 - i
\]

to:

\[
\text{turn} = (\text{turn} + 1) \mod n
\]

where \(n\) is the number of processes.
Evaluate the algorithm when the number of concurrently executing processes is greater than two.

Problem 2 – 5.10 in 9th edition
A software approach to mutual exclusion is Lamport’s bakery algorithm. The algorithm is as follows:

```java
boolean choosing[n];
int number[n];
while (true) {
    choosing[i] = true;
    number[i] = 1 + getmax(number[], n);
    choosing[i] = false;
    for (int j = 0; j < n; j++){
        while (choosing[j]) { }
        while ((number[j] != 0) && (number[j],j) < (number[i],i)) { }
    }
    /* critical section */;
    number [i] = 0;
    /* remainder */;
}
```

The arrays \textit{choosing} and \textit{number} are initialized to false and 0, respectively. The ith element of each array may be read and written by process i, but only read by any other process.

Also, the notation \((a,b) < (c,d)\) is defined as \([a < c \text{ or } (a = c \text{ and } b < d)]\).

1. Describe the algorithm in your own words.
2. Show that it enforces mutual exclusion and avoids deadlocks.
Problem 3 – 5.21 in 9th edition

The following problem was once used on an exam:

Jurassic Park consists of a dinosaur museum and a park for safari riding. There are \( m \) passengers and \( n \) single-passenger cars. Passengers wander around the museum for a while, then line up to take a ride in a safari car. When a car is available, it loads the one passenger it can hold and rides around the park for a random amount of time. If the \( n \) cars are all out riding passengers around, then a passenger who wants to ride waits; if a car is ready to load but there are no waiting passengers, then the car waits. Use semaphores to synchronize the \( m \) passenger processes and the \( n \) car processes.

The following skeleton code was found on a scrap of paper on the floor of the exam room. Grade it for correctness. Ignore syntax and missing variable declarations. Remember that \( P \) and \( V \) correspond to \texttt{semWait} and \texttt{semSignal}.

```plaintext
resource Jurassic_Park()
    sem car_avail := 0, car_taken := 0, car_filled := 0, passengerReleased := 0
process passenger(i := 1 to num_passengers)
    do true -> nap(int(random(1000*wander_time)))
        P(car_avail); V(car_taken); P(car_filled)
        P(passengerReleased)
    od
end passenger

process car(j := 1 to num_cars)
    do true -> V(car_avail); P(car_taken); V(car_filled)
        nap(int(random(1000*ride_time)))
        V(passengerReleased)
    od
end car
end Jurassic_Park
```

Problem 4 – 6.6 in 9th edition

In the code below, three processes are competing for six resources labeled A to F.

1. Using a resource allocation graph, show the possibility of a deadlock in this implementation.
2. Modify the order of some of the get requests (within the procedures themselves, don’t move code across procedures) to prevent the possibility of any deadlock.
Problem 5 – 6.14 in 9th edition

Suppose the two processes below are running concurrently, and that they share the semaphore variables S and R, initialized to 1, and the integer variable x, initialized to 0.

Could the concurrent execution of these processes result in one of them being blocked forever? If yes, what is the execution sequence that would cause this. If no, explain why not.

Problem 6 – 6.15 in 9th edition

Consider a system with 4 processes and a single resource. The current state of the claim and allocation matrices are shown below. What are the minimum number of units of the resource needed to be available for this state to be safe?

\[
C = \begin{pmatrix} 3 \\ 2 \\ 9 \\ 7 \end{pmatrix}, \quad A = \begin{pmatrix} 1 \\ 1 \\ 3 \\ 2 \end{pmatrix}
\]
Part 2 - Pair-Programming problems

Problem 7 – 5.21 in 9th edition

Write a program that would correctly solve the Jurassic park problem described in Problem 3 above.

Deliverables:

- Code in a file named jurassic.[language]
- A report file named Jurassic_readMe.txt, in which you explain how you tested the code.

Problem 8

Consider the problem of servicing customers at McRonald’s, a very popular fast food chain in your neighborhood. McRonald’s is known for its delicious burgers at a fair price, and quick service!

The great service is due to the fact that cooks in the restaurant always pre-prepare the burgers, and customers can just pick up their burgers and check out. To be more specific, a customer goes into the restaurant, and heads to the service counter. If a burger is available, the customer picks it up, and then goes to the cashier to pay for it. If no burgers are available, the customer waits until the cooks prepares a new burger.

Cooks prepare burgers only if there is an available spot for them on the service counter. The service counter can hold up to 20 burgers, and if the service counter is full, the cook stops making burgers until a spot opens up.

Write a program that would simulate a typical day at McRonald’s. This code should explicitly indicate the sequence of actions to be performed by a customer and a cook in McRonald’s. For simplicity, assume that there is only one type of burger sold at this branch. You can use as many semaphore variables as you need, but perfect grades will only be given to the logic using the minimum number of semaphores.