**CSC8002 Advanced Programming**

**Assessed Coursework 2**

Suppose a Semaphore class was defined as follows:

public class Semaphore {

protected int value;

public Semaphore()

{

value = 0;

}

public Semaphore(int initial)

{

value = (initial>=0) ? initial : 0;

}

public synchronized void P() throws

InterruptedException

{

while (value==0)

{

wait();

}

value--;

}

public synchronized void V()

{

value++;

notify();

} }

**Exercise 1:** A BinarySemaphore class

Derive a BinarySemaphore class from the Semaphore class defined as above. Notice that value was declared protected in the Semaphore class, so you can use it directly in any class that extends this class. Remember that while writing the definition of the constructor of a subclass, if you want to call a constructor of the superclass you must make this call the first statement. If your call needs to be conditional you might find the following construct, called a conditional expression, useful: expression1 ? expression2 : expression3 The conditional expression is evaluated as follows: if the boolean expression1 evaluates to true, the result of the conditional expression is expression2, otherwise the result is expression3. **Exercise 2:**Using semaphores

Write a program that creates four threads. The first one repeatedly prints the letter “W”, the second one “X”, the third one “y”, and the fourth one “z”. Using binary and general purpose counting semaphores make the threads to synchronise so that the output of the program satisfies the following conditions.

 “W”s and “X”s must alternate in the output string and the first “W” must

precede the first “X”.

 “y”s and “z”s must alternate in the output string and the first “y” must precede

the first “z”.

 The total number of “y”s and “z”s that have been output at any given point in

the output string cannot exceed the number of “W”s that have been output up

to that point.

To make the program output more varied sequences of letters each time you run it, make the threads to sleep a random number of milliseconds before printing each letter. To do so you can use the random() method from the Math class which returns a double value, greater than or equal to 0.0 and less than 1.0. To stop the threads the main program can simply use System.exit(0) as its last statement. **Do not use the interrupt() method at all in your program.**