Johan van der Veen

Lab4

1) **What happens? Is “Hello” ever printed? Why or why not? Why does the program appear to stall?**

The program stalls and never prints hello. Because the semaphore is initially set to zero when

wait command is issued it cannot decrement the semaphore anymore and therefore waits for a

signal that it will never receive.

Task1:

from pythonThreading import \*

aSemaphore = Semaphore(0)

aSemaphore.signal();

aSemaphore.wait()

print "hello"

2) **In addition to adding a line of code to your python file, how else could you modify the code (without removing the wait()) so that the “Hello” is printed?**

With out adding a line of code I could simply set the initial value of the semaphore to

1 instead of 0.

3)**Q3 : What behavior do you notice? What is the order that function\_1 and function-2 are issued in the code, and what is the order that they finish? Why might the first thread that issues function\_1 complete last?**

Since both functions are issued concurrently we expect the one with less operations to complete

first. This is what we saw, function 2 completed and printed first because it performs about 100

times less operations.

Task2:

from pythonThreading import \*

def function\_1 (val\_1, val\_2,val\_3):

value = val\_3

for x in range(1,val\_1):

for y in range (1,val\_2):

value = value +x +y

print "function 1 says:", value

semaphore.signal()

def function\_2(val\_1,val\_2,val\_3):

value = val\_3

for x in range(1,val\_1):

for y in range(1,val\_2):

value = value +x +y

semaphore.wait()

print "function 2 says:", value

semaphore = Semaphore(0)

Thread(function\_1, 1000,1000,3.6)

Thread(function\_2, 100, 100, 3.6)