CS350 Assignment 7

Chris-Emio (chrisr98@bu.edu)

April 16, 2020

Written Part

Problem 1

a)

Task	Period (ms)	Job Length (ms)	utilization
T_1	6	1	0.167
T_2	12	2	0.167
T_3	16	2	0.125
T_4	18	3	0.167
T_5	15	1	0.067
T_6	10	4	0.4
T_7	8	4	0.5

b) Sum of utilization is $0.167 + 0.167 + 0.125 + 0.167 + 0.067 = 0.693$ This task set would be schedulable under EDF because 0.693 is less $\frac{\beta \times N+1}{\beta+1}$ given that N = 1 and $\beta \approx 5$.

c) Sum of utilization is $0.167+0.167+0.125+0.167+0.067+0.4+0.5 = 1.593$. This new task set would be schedulable under EDF because 1.593 is still $\frac{\beta \times N+1}{\beta+1}$ given that N = 1 and $\beta \approx 5$.

d) This task would be schedulable under G-EDF because the sum of utilization is still less than N = 2.

e) Under the new algorithm the task-set would not be schedulable.

Problem 2

a)

```
semaphore m = 1
//Repeat for N students
Process student:
    //Wait until copy is available
    wait(m)
    //Student reads for h minutes
    //Once done students take break for k minutes and passes the copy
    signal(m)
forever
```

b)

```
semaphore m = 1
T = 0
//Repeat for N students
Process student:
    if (student had document for T time) {
        wait(B)
        reset T
    } else {
        //Wait until copy is available
        wait(m)
        //Student reads for h minutes
        //Add time to T
        //Once done students take break for k minutes and passes the copy
        signal(m)
    }
forever
```

I don't think T would be a semaphore here because it does signal for when the resource is available.

Problem 3

a) Assuming that chef will continuously add 1 sushi to the plate if there is at least 1 space so customer doesn't have to wait for chef to cook, and this is for 1 chef and 1 customer

```
semaphore chef's special = 0
semaphore sushi = 1
Process customer:
    //go in and grab a seat and order
    signal(chef's special)

    //if customer doesn't want anymore
    wait(chef's special)
forever

Process chef:
    wait(chef's special)

    if(plate is full) {
        wait(sushi)
    } else {
        signal(sushi)
    }
forever
```

b) Same assumption as before. And assuming that all customers who order the chef's special eat from the plate it and there are M chefs.

```
semaphore chef's special = 0
```

```
semaphore plate = 1
semaphore M = 1 (number of chefs)
Process customer:
    //go in and grab a seat and order
    wait(M)
    if (chef's available) {
        signal(M)
    } else {
        //if customer doesn't want anymore
        //or all chefs are busy
        wait(M)
    }
    signal(chef's special)
forever

Process chef:
    wait(chef's special)
    if(plate is full) {
        wait(sushi)
    } else {
        signal(sushi)
    }
forever
```