CS 109: Introduction to Computing
Goodney
Fall 2017

Homework Assignment 3

Assigned: 10/16/2017 via Blackboard
Due: 2359 (i.e. 11:59:00 pm) on 10/30/2017 via Blackboard

Notes:

a. This is the third homework assignment of CS 109. It is worth 7.5% of your overall grade for this class. You must solve all problems correctly to obtain full credit.

b. You are free to discuss the problems on this assignment with your classmates. However, to receive credit, you must write up and submit solutions completely on your own. You are responsible for understanding your answers. The purpose of discussing the questions with your classmates is to deepen your understanding of the material – not simply to obtain answers from them. To get the most out of the class, you are strongly encouraged to make a serious effort to understand and solve the problems on your own before discussing them with others.

c. On the work you turn in, you must list the names of everyone with whom you discussed the assignment.

d. All answers must be typed, not handwritten. The homework must be turned in as a single **PDF** document on Blackboard. Other formats will not be graded.

Problem 1: Programming [1 point]

Consider the following program:

```
N = 1000
b = 2
a = 0
n = N
while n >= b
        n = n/b
        a = a+1
print a
```

(a) What value will the above program print?

(b) What value would be printed if N were initialized to 1024*1024*1024 (instead of 1000)?

(c) What function of N does the variable a compute?

Problem 2: Programming [2 points]

Consider the following pseudo code where a function has been defined for a non-negative integer x. Note that the function calls itself (within the while loop) making it a recursive function.

```
def func(x):
    i = 0
    print x
    while i < x:
        i = i+1
        func(x-1)
```

(a) How many times does the program print if we call `func(0)` and `func(1)`, respectively?

(b) Suppose $T(1)$ represents the number of times the function prints when called with an argument 1, $T(2)$ represents the number of times the function prints when called with an argument 2, … , $T(n-1)$ represents the number of times the function prints when called with an argument $n-1$, and $T(n)$ represents the number of times the function prints when called with an argument n.

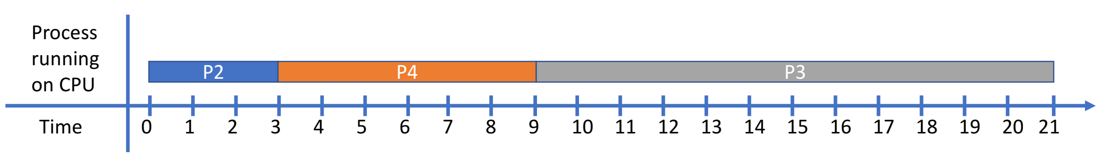
We can now make the following argument: `func(n)` will print once (it prints the number n) and will follow this up by calling `func(n-1)`, n times. According to our notation, each such call prints $T(n-1)$ times. Therefore $T(n) = 1 + nT(n-1)$. Based on this information, does the program run in polynomial time? Provide an explanation for your answer.

Problem 3: Operating Systems [4.5 points]

Consider this table when answering the following questions about scheduling. The table shows a list of processes P1 through P5. For each process, the total running time is given in the table. For example, if process P3 was given the CPU at time t, and was allowed to run to completion without being interrupted, it would finish at time (t + 12).

Process	Total Running Time
P1	**26**
P2	**3**
P3	**12**
P4	**6**
P5	**18**

A schedule for tasks can be shown as a chart. For instance, if processes P2, P4, P3 arrive in that order and the OS processes them using a first come, first served schedule, the schedule of tasks would be as follows:



Note that in the above schedule, process P2 does not have to wait. It gets the CPU immediately (i.e at time = 0). Process P4 has to wait for 3 units of time (while process P2 finishes) and process P3 has to wait 9 units of time (while P2 and P4 finish).

(a) Suppose the processes arrive in the order P1, P2, P3, P4, P5. Draw a chart for this schedule of tasks assuming first come, first served processing.

(b) Find the wait time for each process in (a) and the average waiting time for the 5 processes according to the schedule in (a).

(c) What order should the 5 processes arrive in if the result of first come, first served is to minimize the overall average wait time?

(d) Give the schedule of tasks in chart form corresponding to (c).

(e) What is the average wait time corresponding to (c)?

(f) Recall that with round robin scheduling, each process is given a small unit of CPU time (time quantum), q. After this has elapsed, the process is removed from the CPU (even though it has not finished executing) and added to the end of the queue. Suppose the processes are initially ordered in the queue as P1, P2, P3, P4, P5 and let q = 6. Give the chart for this schedule. Find the wait time for each process and the average wait time.

(g) Suppose that there are n processes in the queue and the time quantum is q. Describe (in terms of n and q) the maximum number of time units a process will wait to use the CPU for the first time. **Hint:** each process receives at most $\frac{1}{n}$ of the CPU time in chunks of at most q time units.

(h) What are the advantages of a short quantum? What are the disadvantages?

(i) What is the purpose of system calls? Why are they necessary?