**Assignment 7 (Simulation of a Printer Operation)**

In this assignment, you write a program to simulate how a printer operates in order to identify the adequacy of a single printer in handling the amount of printing tasks it receives during a given time period. What do you need to know in order to do a simulation? The inputs are: simulation duration (total number of time units), printer speed (number of printed pages per minute, also known as *page-rate*), the size of a printing task (anywhere between 1 page and *maximum* number of pages expected – a random number), and the frequency a printing task may be requested (anywhere between *minIAT* and *maxIAT*). What does simulation produce? The outputs are: the average wait time, number of tasks processed, number of tasks still waiting in the queue when simulation time is over.

Printing is a complex, but for the sake of this assignment, we reduce the complexity to a minimum. Assume that there is no printing task interruption, and there are no task withdrawals. Besides, how to print is irrelevant, thus only the needed time to complete a task is tracked.

As we mentioned in class that there are different ways to do simulations. You can mimic the example in the Chapter 5 to complete the assignment with perhaps only minor modifications. However, if you want to do it in a different fashion, *read on*…..

**Let’s start with the design of a printing task; what would be its properties and behaviors?**

A printing task has the number of pages to be printed (minimum 1, but not exceeding *maxNumPages*). A task also has a time stamp when it is received by the printer (and gets enqueued). A task is a passive object, thus it doesn’t do much other than the following:

**Getters and setters**

**int waitTime(int currentTime):** it returns the wait time given the current time.

**Next, we look at design of a printer object:**

What would a printer need to “know” minimally in order to do its job? It knows its printing speed (number of pages printed per minute, aka: page-rate). It also knows whether or not the current printing task is finished. So, a printer object may have the following responsibilities:

**void tick():** it simulates the passage of a second to move a printing job forward.

**boolean isBusy():** returns true if the current task is not finished, and false otherwise.

**void startNext(Task newTask):** The “new task” becomes the “current task”, and it “figures out” the time needed (given the page-rate) to complete the task.

**Finally, let’s figure out how to proceed the simulation by mimicking the “real world”:**

1. At the start, you need a “printer” instance, and surely, a queue (of tasks).
2. Define other needed variables.
3. Given a simulation period in seconds, say, 10,000 seconds (about 167 minutes), you need a loop with an increment of a second. What does it do in any given second?
   1. *It “detects” whether there is a new printing task requested.* What does it mean by “detect”? Well, you pick a random number with “ceiling” maxIAT (maximum inter-arrival time), if it is in fact greater than (or equal to) minIAT (minimum inter-arrival time), then you assume that there is a printing task requested. The values of *maxIAT and minIAT* can be altered so that different frequencies can be considered (which is the power of simulation).
   2. If there is a printing task requested, create a task instance (with the information: the time when the task is received, and the number of pages to be printed – another random number), and enqueue the object.
   3. If there isn’t a printing task requested, then
      * 1. If the printer is busy (there is still a positive remainder of the time needed for the current task), printer keeps “ticking” (to reduce the required time by another second).
        2. If the printer is not busy, and the queue is not empty, get a task from the queue (this becomes the current task), update simulation data, and continue “ticking” on the current task.
4. When the loop is finished, you print the simulation information:

* *The simulation parameters used:* simulation duration (number of total seconds), printer speed (page-rate), values of minIAT and maxIAT, maximum size of a printing task (maximum number of pages), and
* *The results:* average wait time, number of tasks processed, number of tasks still waiting.