OPERATING SYSTEMS 2
Assignment 3 Report

Name : SONGA KOTESH SATVIK

Roll No : CS20BTECH11048

The executables takes the inputs from the file named “input.txt” and creates the file RM-Log.txt , RM-Stats.txt, EDF-Log.txt and EDF-Stats.txt which contain the desired output.

- A class process is declared which represents a process. It contains the processId, processing time, period, deadline, number of times the process should be executed.
- Additional variables declared are
    1. flag – is 1 if the process is ready to be executed or is being executed and 0 when the process is not ready (not in the waiting queue)
    2. rt – is the remaining running time of the process
    3. rep – number of times the process has been executed
    4. wt – waiting time of the process
- The value of the number of process is given in the input, we get the value of n from the input.
- An array of type process and size n is declared to store the n processes.

Rate-Monotonic

- An array x of size n and type process is declared and the processes are filled in the order of their priority (first is highest priority i.e. less processing time).

- We declare variables, time = 0, deadline_misses = 0, prev_k (stores the rep value of the process running in the previous unit of time), index which stores the index of the previous process.
- We create an infinite loop such that it breaks the loop when all the process have been executed.
- We find the process whose flag is 1 in the order of the priority of the processes and reduce the remaining time of the process by 1. If the remaining time is 0, we do
    1. Flag = 0
    2. Remaining time = processing time
    3. Rep++
- Then ,we store the index value and the rep value of the present process.
- We increase the value of time by time++
- We check whether a process is in the waiting queue, if a process is in the waiting queue its waiting time is increased by 1.
- If a process has to be added to the system at that time, the flag of that process is made 1.
  If the flag of the process is already 1 the process is terminated as it misses the deadline, we do rep++, remaining time = processing time, and keep the flag as 1.
- We check whether the flags of all the processes are non-zero, if the flag of all the processes are zero and all the process have been completed, it exits from the loop
- The required information are printed to the files RM-Log.txt and RM-Stats.txt.

Earliest Deadline First

- An array x[n] is declared which is a duplicate of the p[n].
- We declare
    1. Time = 0
    2. Index
    3. Prev_index – stores the previous index
    4. Prev_k – stores the rep value of the process running in the previous unit of time
    5. Deadline_misses = 0

- We create a function which returns the index of the process which has the earliest deadline.
- We run the process of that index i.e, decrease the rt. If the rt is 0,
    1. Flag = 0
    2. Remaining time = running time
    3. Rep++
- We store the index values and the k value
- Increase the time by time++
- We check whether a process is in the waiting queue, if a process is in the waiting queue its waiting time is increased by 1.
- If a process has to be added to the system at that time, the flag of that process is made 1 and the deadline is updated.
  If the flag of the process is already 1 the process is terminated as it misses the deadline, we do rep++, remaining time = processing time, and keep the flag as 1. We update the deadline of the process.
- We check whether the flags of all the processes are non-zero, if the flag of all the processes are zero and all the process have been completed, it exits from the loop
- The required information are printed to the files EDF-Log.txt and EDF-Stats.txt.

PROCESS ID	PROCESSING TIME	PERIOD	REPETITIONS
1	20	80	10
2	30	50	10
3	25	60	10
4	20	100	10
5	35	120	10
6	15	90	10
7	30	100	10
8	25	95	10
9	40	150	10
10	35	130	10