King Saud University
College of Computer and Information Sciences
Computer Science Department

CSC 227 - Operating Systems 2nd Semester 1440-1441 H

Multiprogramming Batch OS-Simulation

Section #52991

Team Members

	NAME	ID
1	Hind Al Qahtani	436200744
2	Alanoud Alotaibi	437200539
3	Shaikha Bin Ateeq	438201519
4	Meaad Bajned	438201250
5	Zaha Alluhaidan	438201298

TA: Afnan Al Gobail

Task Distribution:

NAME	TASK / CLASSES
Hind Al Qahtani	CPU, Hardware, Job, JobCompareSize, Main, OperatingSystem, PCB,
Alanoud Alotaibi	CPU, Hardware, OperatingSystem
Shaikha Bin Ateeq	Main, Job,
Meaad Bajned	OperatingSystem, JobCompareSize
Zaha Alluhaidan	PCB, JobState, TerminationType

Execution Instructions:

job scheduling we use

. In this simulation program of a simple multiprogramming batch operating system, we have hardware specifications assumed as follow: single CPU, hard disk with 2GB, RAM with 192MB for user programs. Our Simulation covers two features of the operating system. First is *job scheduling* follows smallest storage requirement (SSR) policy, the second is *process scheduling* that follows the First-Come, First-Served (FCFS) scheduling algorithm policy. There will be no external input enter the program, in the beginning of simulation, we generate a number of jobs which is 10000000 and we check the requirement that the expected CPU size for each process and expected memory requirement for each job are generated randomly with a uniform distributed between 16 and 512 units and between 16 KB and 256 KB. We also check the hard disk size before add the job is it accommodate a new job or not. After we add all the NEW jobs, we arrange them by follows the smallest storage requirement and added them into a long term scheduling queue. Then we start adding jobs to RAM to start executing the simulation program. We check if the hard disk is not empty then we take first process and execute it directly since we follow first come first serve algorithm. At this point when the job is getting process its state being RUNNING instead of NEW, we allow process to execute until it terminate. Then we generate an interrupt randomly with the probabilities given 10% for process terminates normally and 5% for process terminates abnormally by using Math.*Random*() method that return a pseudorandom double type number greater than or equal to 0.0 and less than 1.0.

. Finally, we print the result of the processed jobs in Result.txt file with the following:

- Total number of jobs processed.
- Average/Minimum/Maximum job size in KB.
- Number of jibs that completed normally.
- Number of jobs completed abnormally.
- Generating report for each terminating process displaying its PID, CUT and termination status.

Reflection on Simulation:

We think the performance generally is good we have two queues, one for long term scheduling and one for short term scheduling, the short term scheduling follows first come first serve scheduling algorithm policy. For this project idea we think read file step is not necessary and effect the performance negatively since that we already have the queue of all jobs that we want to process, we don’t need to read jobs data from the Job.TXT file.

References:

Silberschatz, A., Galvin, P. and Gagne, G., n.d. *Operating System Concepts*.

Part 2: Teamwork					
Criteria	**Student 1**	**Student 2**	**Student 3**	**Student 4**	**Student 5**
Work division: Contributed equally to the work					
Peer evaluation: Level of commitments (Interactivity with other team members), and professional behavior towards team & TA					
Project Discussion: Accurate answers, understanding of the presented work, good listeners to questions					
Time management: Attending on time, being ready to start the demo, good time management in discussion and demo.					
Total/4	0	0	0	0	0