يوسف محمد محمود الرجال --7806

package projectOs;

import java.util.Random;

import java.util.\*;

public class Main {

public static void main(String[] args) {

Queue<process> all\_processes\_unsorted = new LinkedList<>();// (lts) queue is playing the role of LongtermSchandular queue (decide which process to enter to cpu)

queue lts = new queue();

queue comming\_processes\_to\_system= new queue();

queue waitting\_to\_EnterFirstQ = new queue(); // i make it have highpriorty than LTS queue (i see that the process already on the system have high prority than the process that didnot enter the system)

queue waitting\_to\_EnterSecondQ = new queue();

queue waitting\_to\_EnterthirdQ = new queue();

int time\_of\_cpu = 128; //assume we will divide every 128 second of cpu

// , (50% for first queue , 30%...,20%...)

queue firstQ\_roundroben = new queue(10,(int)(time\_of\_cpu\*50/100));

queue secondQ\_roundroben = new queue(20,(int)(time\_of\_cpu\*30/100));

queue thirdQ\_fcfs = new queue(30,(int)(time\_of\_cpu\*20/100));

Random predict = new Random();

get\_random\_unique\_time generate\_random =new get\_random\_unique\_time();

// generate the 100 process and push them to lts queue

for (int id = 1; id <= 100; id++) {

//assume all processes arrives in random time in the first 500 seconds and no two processes arrives in the same time

int arrival\_time = generate\_random.get\_unique\_arrrival\_time();

process p = new process(id, predict.nextInt(100) + 1,arrival\_time);

all\_processes\_unsorted.add(p);

// System.out.println("id"+p.id+ " time "+ p.time\_remaning + " arrival time" + p.getArrival\_time());

}

// sorted processes

List<process> sortedProcesses = new ArrayList<>(all\_processes\_unsorted);

sortedProcesses.sort(Comparator.comparingInt(process::getArrival\_time));

comming\_processes\_to\_system.add\_all\_process\_sorted(sortedProcesses);

// Print the sorted list

// System.out.println("\nSorted List based on Arrival Time:");

// for (process p : comming\_processes\_to\_system.my\_queue) {

// System.out.println(p);

// }

// start of the system

int number\_of\_finshed\_processes = 0;

int system\_time=0;

while (number\_of\_finshed\_processes != 100){

// if the all queues are empty set the System time to the min arrival time

if (lts.is\_queue\_empty()

&& firstQ\_roundroben.is\_queue\_empty() && secondQ\_roundroben.is\_queue\_empty() && thirdQ\_fcfs.is\_queue\_empty()

&& waitting\_to\_EnterFirstQ.is\_queue\_empty() && waitting\_to\_EnterSecondQ.is\_queue\_empty() && waitting\_to\_EnterthirdQ.is\_queue\_empty()){

system\_time=comming\_processes\_to\_system.get\_first\_element().arrival\_time;

}

// lw fyh new commers hd5lhom lts bs (msh hd5lhom el first queue)

checkForNewComers.checkForNewCommers(comming\_processes\_to\_system,system\_time,lts);

// if there is some free spaces in first queue then push to it some process from lts or from waitingQ

while (!firstQ\_roundroben.is\_queue\_full() && (!lts.is\_queue\_empty() ||!waitting\_to\_EnterFirstQ.is\_queue\_empty())){

if (!waitting\_to\_EnterFirstQ.is\_queue\_empty()){ // the prcess already in the system and havebeen demoted

// i make them have high priorty than the processes doesnot enter the system

waitting\_to\_EnterFirstQ.get\_first\_element().logger("-enter the first queue from waiting queue at time "+ system\_time);

firstQ\_roundroben.enqueue(waitting\_to\_EnterFirstQ.dequeue());

}

else if (!lts.is\_queue\_empty()){

lts.get\_first\_element().logger("-enter the first queue from the long term schadular queue at time " + system\_time);

firstQ\_roundroben.enqueue(lts.dequeue());

}

}

// Now first queue will take time on cpu

int time\_taken\_by\_firstQ\_on\_cpu\_tillnow = 0;

//if there is any element we will do a roundroben with Q=8 with each element till its time on cpu ends

while (!firstQ\_roundroben.is\_queue\_empty() && time\_taken\_by\_firstQ\_on\_cpu\_tillnow < firstQ\_roundroben.getTime\_avalible\_on\_cpu()){

process p = firstQ\_roundroben.dequeue();

if (p.get\_time()<=8){

time\_taken\_by\_firstQ\_on\_cpu\_tillnow+=p.get\_time();

p.logger("-excute "+p.get\_time()+" sec on the first Queue from "+system\_time+" till "+ (system\_time+p.get\_time()));

system\_time+=p.get\_time();

number\_of\_finshed\_processes ++;

p.i\_am\_done(1);

}else{

time\_taken\_by\_firstQ\_on\_cpu\_tillnow+=8;

p.logger("-excute "+"8"+" sec on the first Queue from "+system\_time+" till "+(system\_time+8)+" and enter the waiting second queue");

system\_time+=8;

p.set\_time(p.get\_time()-8);

waitting\_to\_EnterSecondQ.enqueue(p);

}

// to handel the case that the queue still have time on cpu ,

// and queue is free from process , so each time w dequeue a process we check for process in lts or in waiting queue

checkForNewComers.checkForNewCommers(comming\_processes\_to\_system,system\_time,lts);

if (!waitting\_to\_EnterFirstQ.is\_queue\_empty()){

waitting\_to\_EnterFirstQ.get\_first\_element().logger("-enter the first queue from watting queue at time "+system\_time);

firstQ\_roundroben.enqueue(waitting\_to\_EnterFirstQ.dequeue());

}

else if (!lts.is\_queue\_empty()){

lts.get\_first\_element().logger("-enter the first queue from lts at time "+system\_time );

firstQ\_roundroben.enqueue(lts.dequeue());

}

}

// if the first queue empty then go to see the second queue

if (firstQ\_roundroben.is\_queue\_empty()){

// Now check if there is some empty spaces in second queue

while (!secondQ\_roundroben.is\_queue\_full() && !waitting\_to\_EnterSecondQ.is\_queue\_empty()) {

// push to it from waiting to enter QUEUE 2

if (!waitting\_to\_EnterSecondQ.is\_queue\_empty()) {

waitting\_to\_EnterSecondQ.get\_first\_element().logger("-enter the second queue from the second waiting queue at time "+system\_time);

secondQ\_roundroben.enqueue(waitting\_to\_EnterSecondQ.dequeue());

}

}

// Now the second Queue will take time on CPU

int time\_taken\_by\_secondQ\_on\_cpu\_tillnow = 0;

while (!secondQ\_roundroben.is\_queue\_empty() && time\_taken\_by\_secondQ\_on\_cpu\_tillnow < secondQ\_roundroben.getTime\_avalible\_on\_cpu()){

process p = secondQ\_roundroben.dequeue();

if (p.get\_time()<=16){

time\_taken\_by\_secondQ\_on\_cpu\_tillnow+=p.get\_time();

p.logger("-execute "+p.get\_time()+" sec on the second queue from "+system\_time+" till "+(system\_time+p.get\_time()));

system\_time+=p.get\_time();

p.i\_am\_done(2);

number\_of\_finshed\_processes++;

}else{// make it waiting to enter third QUEUE or demote it to be waiting to enter the first QUEUE

time\_taken\_by\_secondQ\_on\_cpu\_tillnow+=16;

// now we have to demote it or promote it

double rand = predict.nextDouble(); // generate a random number between 0 and 1

if (rand < 0.5){ //demote it حنزلها ل كيو 1

p.logger("-execute 16 sec on the second queue "+"from " +system\_time+" till "+(system\_time+16)+"and demoted to first queue ");

waitting\_to\_EnterFirstQ.enqueue(p);

}else{// promote it حنرئيها لكيو 3 بس بعد منرنها علي كيو 2

p.logger("-execute 16 sec on the second queue "+"from " +system\_time+" till "+(system\_time+16)+"and prmoted to the third waiting queue ");

waitting\_to\_EnterthirdQ.enqueue(p);

}

system\_time+=16;

p.set\_time(p.get\_time()-16);

}

// to handel the case that Queue 2 is empty and it still have time on cpu

// every time we deqeue an element we insert another element if some one is waiting

if (!waitting\_to\_EnterSecondQ.is\_queue\_empty()) {

waitting\_to\_EnterSecondQ.get\_first\_element().logger("-enter the second queue from the second waiting queue at time"+system\_time);

secondQ\_roundroben.enqueue(waitting\_to\_EnterSecondQ.dequeue());

}

// 3shan lw fyh elemnts wslt l first hwa msh hyd5ol f third (fkk mn comment da)

// hwa str gy mlhosh lzma bs hsebo akni shghal tread showya , // every time we change system time we check for new arrivals

checkForNewComers.checkForNewCommers(comming\_processes\_to\_system,system\_time,lts);

}

}

// if the first and second queue are empty then check the third queue

if (firstQ\_roundroben.is\_queue\_empty() && secondQ\_roundroben.is\_queue\_empty()){

// check if there is some free spaces on third queue then push to it some process from waiting Q

while (!thirdQ\_fcfs.is\_queue\_full() && !waitting\_to\_EnterthirdQ.is\_queue\_empty()){

if (!waitting\_to\_EnterthirdQ.is\_queue\_empty()){

waitting\_to\_EnterthirdQ.get\_first\_element().logger("-enter the third queue at time "+system\_time);

thirdQ\_fcfs.enqueue(waitting\_to\_EnterthirdQ.dequeue());

}

}

//Fcfs queue is on cpu

int time\_taken\_by\_thirdQ\_on\_cpu\_tillnow = 0;

while (!thirdQ\_fcfs.is\_queue\_empty() && time\_taken\_by\_thirdQ\_on\_cpu\_tillnow < thirdQ\_fcfs.getTime\_avalible\_on\_cpu()){

process p = thirdQ\_fcfs.dequeue(); // tl3tha bra el system

p.logger("-excute "+p.get\_time()+" on the third queue from "+system\_time +" till "+(system\_time+p.get\_time()));

p.i\_am\_done(3);

number\_of\_finshed\_processes++;

time\_taken\_by\_thirdQ\_on\_cpu\_tillnow += p.get\_time();

system\_time+=p.get\_time();

// to handel the case that Queue 3 is empty and it still have time on cpu

// every time we deqeue an element we insert another element if some one is waiting

if (!waitting\_to\_EnterthirdQ.is\_queue\_empty()){

p.logger("-enter the third queue at time "+system\_time);

thirdQ\_fcfs.enqueue(waitting\_to\_EnterthirdQ.dequeue());

}

// every time we change system time we check for new arrivals , akni shgal thread

checkForNewComers.checkForNewCommers(comming\_processes\_to\_system,system\_time,lts);

}

}

// System.out.println("NO FINISHED PROCESSES "+ number\_of\_finshed\_processes);

}

// System.out.println(system\_time);

}

}