CS-266 Mid Sem Lab Exam

Roll no:- 201951134 Name:- Sameer Anand

**Code:-**

**Output:** import java.util.\*;

public class preendsem {

    static void find\_avgtime\_rr(int proc[], int n, int bt[], int wt[], int quantum) {

        int temp\_wt[] = new int[n];

        for (int i = 0; i < n; i++) {

            bt[i] -= quantum;

            if (i != 0) {

                temp\_wt[i] = temp\_wt[i - 1] + quantum;

                wt[i] = wt[i - 1] + quantum;

            }

        }

        findavgwatingtimeSJr(proc, n, bt, wt);

        System.out.println("Processes " + " Burst time " + " Waiting time " + " Turn around time");

        int total\_wt = 0, total\_tat = 0;

        int tat[] = new int[n];

        for (int i = 0; i < n; i++) {

            total\_wt = total\_wt + wt[i];

            total\_tat = total\_tat + tat[i];

            System.out.println("      " + (i + 1) + "        " + bt[i] + "      " + temp\_wt[i] + "       " + tat[i]);

        }

    }

    static void findavgwatingtimeSJr(int proc[], int n, int bt[], int wt[]) {

        int rt[] = new int[n];

        for (int i = 0; i < n; i++)

            rt[i] = bt[i];

        int complete = 0, t = 0, minm = Integer.MAX\_VALUE;

        int shortest = 0, finish\_time;

        boolean check = false;

        while (complete != n) {

            for (int j = 0; j < n; j++) {

                if ((rt[j] < minm) && rt[j] > 0) {

                    minm = rt[j];

                    shortest = j;

                    check = true;

                }

            }

            if (check == false) {

                t++;

                continue;

            }

            rt[shortest]--;

            minm = rt[shortest];

            if (minm == 0)

                minm = Integer.MAX\_VALUE;

            if (rt[shortest] == 0) {

                complete++;

                check = false;

                finish\_time = t + 1;

                wt[shortest] += finish\_time - bt[shortest];

                if (wt[shortest] < 0)

                    wt[shortest] = 0;

            }

            t++;

        }

    }

    static void find\_turnaroundtime(int processes[], int n, int bt[], int wt[], int tat[]) {

        for (int i = 0; i < n; i++) {

            tat[i] = bt[i] + wt[i];

        }

    }

    static void FAT(int processes[], int n, int bt[], int quantum) {

        int wt[] = new int[n], tat[] = new int[n];

        int total\_wt = 0, total\_tat = 0;

        find\_avgtime\_rr(processes, n, bt, wt, quantum);

        find\_turnaroundtime(processes, n, bt, wt, tat);

        System.out.println("Processes " + " Burst time " + " Waiting time " + " Turn around time");

        for (int i = 0; i < n; i++) {

            total\_wt = total\_wt + wt[i];

            total\_tat = total\_tat + tat[i];

            System.out.println(" " + (i + 1) + " " + bt[i] + " " + wt[i] + " " + tat[i]);

        }

        System.out.println("Average waiting time = " + (float) total\_wt / (float) n);

        System.out.println("Average turn around time = " + (float) total\_tat / (float) n);

    }

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        int n, q;

        System.out.println("Enter no of process");

        n = sc.nextInt();

        int processes[] = new int[n];

        int burst\_time[] = new int[n];

        for (int i = 0; i < n; i++) {

            System.out.print("Enter process id for process" + (i + 1) + " : ");

            processes[i] = sc.nextInt();

            System.out.print("Enter burst time for process" + (i + 1) + " : ");

            burst\_time[i] = sc.nextInt();

        }

        System.out.println("Enter name without roll spaces");

        String s = sc.next();

        System.out.print("Enter your roll number : ");

        int roll = sc.nextInt();

        q = GQ(s, roll);

        System.out.println("Quantum is =  " + q);

        FAT(processes, n, burst\_time, q);

    }

    public static int GQ(String s, int roll) {

        int ans = get\_ascii(s);

        ans %= 20;

        if (ans != 0) {

            return ans;

        } else {

            ans += get\_rollno(roll);

            ans %= 20;

            if (ans != 0) {

                return ans;

            } else {

                return (int) (Math.random() \* 19 + 1);

            }

        }

    }

    public static int get\_rollno(int m) {

        int n, sum = 0;

        while (m > 0) {

            n = m % 10;

            sum = sum + n;

            m = m / 10;

        }

        return m;

    }

    public static int get\_ascii(String s) {

        int sum = 0;

        for (int i = 0; i < s.length(); i++)

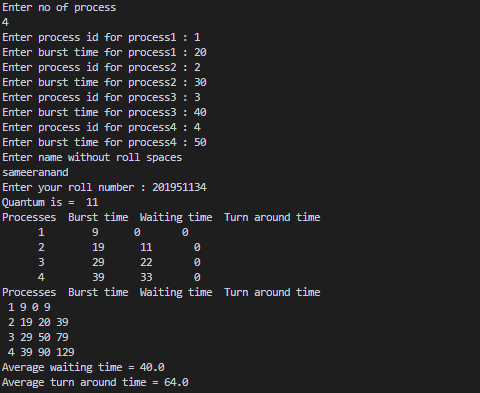
            sum += s.charAt(i);

        return sum;

    }

}

**-**

****