Experiment Number: 8

Problem Statement:Write a program to compute the finish time, turnaround time and waiting time for the following algorithms:

a) First come First serve b) Shortest Job First (Preemptive and Non-Preemptive)

c) Priority (Preemptive and Non-Preemptive) d) Round robin

NAME: Manoj Dhanraj MuleROLLNO: 71

CLASS: IT-B BATCH: B3

DATE OF PERFORMANCE: 15/10/2024

#include <stdio.h> #include <stdlib.h>

typedef struct {

int pid, at, bt, prio, st, ct, wt, tat, rt, remaining\_bt;

} Process;

void FCFS(Process \*proc, int n) { int current\_time = 0;

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

proc[i].st = (current\_time < proc[i].at) ? proc[i].at : current\_time; proc[i].ct = proc[i].st + proc[i].bt;

proc[i].tat = proc[i].ct - proc[i].at;

proc[i].wt = proc[i].tat - proc[i].bt; current\_time = proc[i].ct;

}

}

void SJF\_NonPreemptive(Process \*proc, int n) {

int completed = 0, current\_time = 0, min\_bt, shortest; int is\_completed[50] = {0};

while (completed != n) { min\_bt = 9999;

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

if (proc[i].at <= current\_time && is\_completed[i] == 0) {

if (proc[i].bt < min\_bt || (proc[i].bt == min\_bt && proc[i].at < proc[shortest].at)) {

min\_bt = proc[i].bt; shortest = i;

}

}

}

if (min\_bt == 9999) { current\_time++;

} else {

proc[shortest].st = current\_time;

proc[shortest].ct = proc[shortest].st + proc[shortest].bt; proc[shortest].tat = proc[shortest].ct - proc[shortest].at; proc[shortest].wt = proc[shortest].tat - proc[shortest].bt; is\_completed[shortest] = 1;

current\_time = proc[shortest].ct; completed++;

}

}

}

void SJF\_Preemptive(Process \*proc, int n) {

int current\_time = 0, completed = 0, shortest = 0;

int min\_bt = 9999, is\_completed[50] = {0}, prev = -1;

while (completed != n) { for (int i = 0; i < n; i++) {

if (proc[i].at <= current\_time && is\_completed[i] == 0 && proc[i].remaining\_bt < min\_bt) {

min\_bt = proc[i].remaining\_bt; shortest = i;

}

}

if (min\_bt == 9999) { current\_time++; continue;

}

if (prev != shortest) {

proc[shortest].st = (current\_time < proc[shortest].at) ? proc[shortest].at : current\_time;

prev = shortest;

}

proc[shortest].remaining\_bt--; current\_time++;

min\_bt = proc[shortest].remaining\_bt;

if (proc[shortest].remaining\_bt == 0) { proc[shortest].ct = current\_time;

proc[shortest].tat = proc[shortest].ct - proc[shortest].at; proc[shortest].wt = proc[shortest].tat - proc[shortest].bt; is\_completed[shortest] = 1;

completed++; min\_bt = 9999;

}

}

}

void Priority\_NonPreemptive(Process \*proc, int n) {

int completed = 0, current\_time = 0, highest\_prio, chosen; int is\_completed[50] = {0};

while (completed != n) { highest\_prio = 9999;

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

if (proc[i].at <= current\_time && is\_completed[i] == 0) {

if (proc[i].prio < highest\_prio || (proc[i].prio == highest\_prio && proc[i].at < proc[chosen].at)) {

highest\_prio = proc[i].prio; chosen = i;

}

}

}

if (highest\_prio == 9999) { current\_time++;

} else {

proc[chosen].st = current\_time;

proc[chosen].ct = proc[chosen].st + proc[chosen].bt; proc[chosen].tat = proc[chosen].ct - proc[chosen].at;

proc[chosen].wt = proc[chosen].tat - proc[chosen].bt; is\_completed[chosen] = 1;

current\_time = proc[chosen].ct; completed++;

}

}

}

void Priority\_Preemptive(Process \*proc, int n) {

int current\_time = 0, completed = 0, highest\_prio, chosen = -1; int is\_completed[50] = {0};

while (completed != n) { highest\_prio = 9999;

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

if (proc[i].at <= current\_time && is\_completed[i] == 0 && proc[i].prio

< highest\_prio) {

highest\_prio = proc[i].prio; chosen = i;

}

}

if (chosen == -1) { current\_time++;

} else {

if (proc[chosen].remaining\_bt == proc[chosen].bt) { proc[chosen].st = current\_time;

}

proc[chosen].remaining\_bt--; current\_time++;

if (proc[chosen].remaining\_bt == 0) { proc[chosen].ct = current\_time;

proc[chosen].tat = proc[chosen].ct - proc[chosen].at; proc[chosen].wt = proc[chosen].tat - proc[chosen].bt; is\_completed[chosen] = 1;

completed++;

}

chosen = -1;

}

}

}

void RoundRobin(Process \*proc, int n, int quantum) {

int current\_time = 0, completed = 0, is\_completed[50] = {0};

while (completed != n) { for (int i = 0; i < n; i++) {

if (proc[i].remaining\_bt > 0 && proc[i].at <= current\_time) { if (proc[i].remaining\_bt == proc[i].bt) {

proc[i].st = current\_time;

}

if (proc[i].remaining\_bt > quantum) { current\_time += quantum; proc[i].remaining\_bt -= quantum;

} else {

current\_time += proc[i].remaining\_bt; proc[i].ct = current\_time;

proc[i].tat = proc[i].ct - proc[i].at;

proc[i].wt = proc[i].tat - proc[i].bt; proc[i].remaining\_bt = 0; completed++;

}

}

}

}

}

void printProcessTable(Process \*proc, int n) { printf("\nPID\tAT\tBT\tCT\tTAT\tWT\n"); for (int i = 0; i < n; i++) {

printf("%d\t%d\t%d\t%d\t%d\t%d\n", proc[i].pid, proc[i].at, proc[i].bt, proc[i].ct, proc[i].tat, proc[i].wt);

}

}

int main() {

int n, choice, quantum;

printf("Enter the number of processes: ");

scanf("%d", &n);

Process proc[50];

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

printf("Enter process %d Arrival Time, Burst Time and Priority: ", i+1); proc[i].pid = i+1;

scanf("%d%d%d", &proc[i].at, &proc[i].bt, &proc[i].prio); proc[i].remaining\_bt = proc[i].bt;

}

printf("Choose the scheduling algorithm:\n");

printf("1. FCFS\n2. SJF (Non-Preemptive)\n3. SJF (Preemptive)\n4. Priority (Non-Preemptive)\n5. Priority (Preemptive)\n6. Round Robin\n");

scanf("%d", &choice);

switch (choice) { case 1:

FCFS(proc, n); break;

case 2:

SJF\_NonPreemptive(proc, n); break;

case 3:

SJF\_Preemptive(proc, n); break;

case 4:

Priority\_NonPreemptive(proc, n); break;

case 5:

Priority\_Preemptive(proc, n); break;

case 6:

printf("Enter the time quantum: "); scanf("%d", &quantum); RoundRobin(proc, n, quantum); break;

default:

printf("Invalid Choice!\n"); return 0;

}

printProcessTable(proc, n);

return 0;

}

**Output:**

**FCFS**

**manoj@manoj-VirtualBox:~/oslab$ gcc scheduling.c -o scheduling**

**manoj@manoj-VirtualBox:~/oslab$ ./scheduling**

**Enter the number of processes: 6**

**Enter process 1 Arrival Time, Burst Time and Priority: 0 9 1**

**Enter process 2 Arrival Time, Burst Time and Priority: 1 3 2**

**Enter process 3 Arrival Time, Burst Time and Priority: 1 2 3**

**Enter process 4 Arrival Time, Burst Time and Priority: 1 4 4**

**Enter process 5 Arrival Time, Burst Time and Priority: 2 3 5**

**Enter process 6 Arrival Time, Burst Time and Priority: 3 2 6**

**Choose the scheduling algorithm:**

**1. FCFS**

**2. SJF (Non-Preemptive)**

**3. SJF (Preemptive)**

**4. Priority (Non-Preemptive)**

**5. Priority (Preemptive)**

**6. Round Robin**

**1**

**PID AT BT CT TAT WT**

**1 0 9 9 9 0**

**2 1 3 12 11 8**

**3 1 2 14 13 11**

**4 1 4 18 17 13**

**5 2 3 21 19 16**

**6 3 2 23 20 18**

**manoj@manoj-VirtualBox:~/oslab$ ^C**

**manoj@manoj-VirtualBox:~/oslab$**

**SJF Non-Preemptive**

**manoj@manoj-VirtualBox:~/oslab$ ./scheduling**

**Enter the number of processes: 5**

**Enter process 1 Arrival Time, Burst Time and Priority: 2 6 1**

**Enter process 2 Arrival Time, Burst Time and Priority: 5 2 2**

**Enter process 3 Arrival Time, Burst Time and Priority: 1 8 3**

**Enter process 4 Arrival Time, Burst Time and Priority: 0 3 4**

**Enter process 5 Arrival Time, Burst Time and Priority: 4 4 5**

**Choose the scheduling algorithm:**

**1. FCFS**

**2. SJF (Non-Preemptive)**

**3. SJF (Preemptive)**

**4. Priority (Non-Preemptive)**

**5. Priority (Preemptive)**

**6. Round Robin**

**2**

**PID AT BT CT TAT WT**

**1 2 6 9 7 1**

**2 5 2 11 6 4**

**3 1 8 23 22 14**

**4 0 3 3 3 0**

**5 4 4 15 11 7**

**manoj@manoj-VirtualBox:~/oslab$**

**SJF Preemptive**

**manoj@manoj-VirtualBox:~/oslab$ ./scheduling**

**Enter the number of processes: 5**

**Enter process 1 Arrival Time, Burst Time and Priority: 2 6 1**

**Enter process 2 Arrival Time, Burst Time and Priority: 5 2 2**

**Enter process 3 Arrival Time, Burst Time and Priority: 1 8 3**

**Enter process 4 Arrival Time, Burst Time and Priority: 0 3 4**

**Enter process 5 Arrival Time, Burst Time and Priority: 4 4 5**

**Choose the scheduling algorithm:**

**1. FCFS**

**2. SJF (Non-Preemptive)**

**3. SJF (Preemptive)**

**4. Priority (Non-Preemptive)**

**5. Priority (Preemptive)**

**6. Round Robin**

**3**

**PID AT BT CT TAT WT**

**1 2 6 15 13 7**

**2 5 2 7 2 0**

**3 1 8 23 22 14**

**4 0 3 3 3 0**

**5 4 4 10 6 2**

**manoj@manoj-VirtualBox:~/oslab$**

**Priority Non-Preemptive**

**manoj@manoj-VirtualBox:~/oslab$ ./scheduling**

**Enter the number of processes: 7**

**Enter process 1 Arrival Time, Burst Time and Priority: 0 3 2**

**Enter process 2 Arrival Time, Burst Time and Priority: 2 5 6**

**Enter process 3 Arrival Time, Burst Time and Priority: 1 4 3**

**Enter process 4 Arrival Time, Burst Time and Priority: 4 2 5**

**Enter process 5 Arrival Time, Burst Time and Priority: 6 9 7**

**Enter process 6 Arrival Time, Burst Time and Priority: 5 4 4**

**Enter process 7 Arrival Time, Burst Time and Priority: 7 10 10**

**Choose the scheduling algorithm:**

**1. FCFS**

**2. SJF (Non-Preemptive)**

**3. SJF (Preemptive)**

**4. Priority (Non-Preemptive)**

**5. Priority (Preemptive)**

**6. Round Robin**

**4**

**PID AT BT CT TAT WT**

**1 0 3 3 3 0**

**2 2 5 18 16 11**

**3 1 4 7 6 2**

**4 4 2 13 9 7**

**5 6 9 27 21 12**

**6 5 4 11 6 2**

**7 7 10 37 30 20**

**manoj@manoj-VirtualBox:~/oslab$**

**Priority Preemptive**

**manoj@manoj-VirtualBox:~/oslab$ ./scheduling**

**Enter the number of processes: 5**

**Enter process 1 Arrival Time, Burst Time and Priority: 0 3 3**

**Enter process 2 Arrival Time, Burst Time and Priority: 1 4 2**

**Enter process 3 Arrival Time, Burst Time and Priority: 2 6 4**

**Enter process 4 Arrival Time, Burst Time and Priority: 3 4 6**

**Enter process 5 Arrival Time, Burst Time and Priority: 5 2 10**

**Choose the scheduling algorithm:**

**1. FCFS**

**2. SJF (Non-Preemptive)**

**3. SJF (Preemptive)**

**4. Priority (Non-Preemptive)**

**5. Priority (Preemptive)**

**6. Round Robin**

**5**

**PID AT BT CT TAT WT**

**1 0 3 7 7 4**

**2 1 4 5 4 0**

**3 2 6 13 11 5**

**4 3 4 17 14 10**

**5 5 2 19 14 12**

**manoj@manoj-VirtualBox:~/oslab$**

**Round Robin**

**manoj@manoj-VirtualBox:~/oslab$ ./scheduling**

**Enter the number of processes: 3**

**Enter process 1 Arrival Time, Burst Time and Priority: 0 10 1**

**Enter process 2 Arrival Time, Burst Time and Priority: 0 5 2**

**Enter process 3 Arrival Time, Burst Time and Priority: 0 8 3**

**Choose the scheduling algorithm:**

**1. FCFS**

**2. SJF (Non-Preemptive)**

**3. SJF (Preemptive)**

**4. Priority (Non-Preemptive)**

**5. Priority (Preemptive)**

**6. Round Robin**

**6**

**Enter the time quantum: 2**

**PID AT BT CT TAT WT**

**1 0 10 23 23 13**

**2 0 5 15 15 10**

**3 0 8 21 21 13**

**manoj@manoj-VirtualBox:~/oslab$**