Operating Systems LAB 11

Simulation of Preemptive Process Scheduling Algorithms

Spring 2024

Submitted by: **Hassan Zaib**

Registration No: **22pwsce2144**

Class Section: **A**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Submitted to:

Engr. Abdullah Hamid

Month Day, Year (23 06, 2024)

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

Objective:

To implement Round robin scheduling we keep the ready queue as a FIFO queue of processes. New processes are added to the tail of the ready queue. The CPU scheduler picks the first process from the ready queue, sets a timer to interrupt after one time quantum and dispatches the process. A small unit of time called a time quantum or time slice is defined. A time quantum is generally from 10 to 100 milliseconds.

The process may have CPU burst of less than one time quantum. In this case the process itself will release the CPU voluntarily .The scheduler will then proceed to the next process in the ready queue. Otherwise if the CPU burst of the currently running process is longer than 1 time quantum, the timer will go off and will cause an interrupt to the operating system. The average waiting time under Round Robin policy is how ever quite long.

Code:

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

void findWaitingTime(int n, int bt[], int wt[], int at[], int quantum) {
    int i, rem_bt[n];
    for (i = 0; i < n; i++) {
        rem_bt[i] = bt[i];
    }
    int t = 0;  // Current time
    // Keep traversing processes in round robin manner until all processes are complete
    while (1) {
        int done = 1;
        for (i = 0; i < n; i++) {
            // If a process hasn't finished its burst time
            if (rem_bt[i] > 0) {
                done = 0;  // There are still processes to complete
                if (rem_bt[i] > quantum) {
                    // Reduce remaining burst time by quantum
                    rem_bt[i] -= quantum;
                    t += quantum;
                } else {
                    // If this is the last cycle for this process
                    t += rem_bt[i];
                    wt[i] = t - bt[i] - at[i];
                    rem_bt[i] = 0;  // Marking process as complete
                }
            }
        }
        // If all processes are complete
        if (done == 1) {
            break;
        }}}
```

```c
void findTurnAroundTime(int n, int bt[], int wt[], int tat[], int at[]) {
    int i;
    for (i = 0; i < n; i++) {
        tat[i] = bt[i] + wt[i];
    }
}

void findAvgTime(int n, int bt[], int at[], int quantum) {
    int wt[n], tat[n], total_wt = 0, total_tat = 0;
    findWaitingTime(n, bt, wt, at, quantum);
    findTurnAroundTime(n, bt, wt, tat, at);
    printf("Process\tBurst Time\tArrival Time\tWaiting Time\tTurn Around Time\n");
    // Calculate total waiting time and turn around time
    int i;
    for (i = 0; i < n; i++) {
        total_wt = total_wt + wt[i];
        total_tat = total_tat + tat[i];
        printf("%d\t\t%d\t\t%d\t\t%d\t\t%d\n", i + 1, bt[i], at[i], wt[i], tat[i]);
    }
    float avg_wt = (float)total_wt / (float)n;
    float avg_tat = (float)total_tat / (float)n;
    printf("Average Waiting Time = %f\n", avg_wt);
    printf("Average Turnaround Time = %f\n", avg_tat);
}

int main() {
    int n;
    printf("Enter Total Process:\t");
    scanf("%d", &n);
    int bt[n], at[n];
    printf("Enter Burst Time and Arrival Time for %d Processes:\n", n);
    int i;
for (i = 0; i < n; i++) {
    printf("Process %d:\n", i + 1);
    printf("Burst Time: ");
    scanf("%d", &bt[i]);
    printf("Arrival Time: ");
    scanf("%d", &at[i]);
}
int quantum;
printf("Enter Time Quantum:\t");
scanf("%d", &quantum);
findAvgTime(n, bt, at, quantum);
return 0;
}
```

Output:

```
Enter Total Process:    4
Enter Burst Time and Arrival Time for 4 Processes:
Process 1:
Burst Time: 5
Arrival Time: 0
Process 2:
Burst Time: 6
Arrival Time: 0
Process 3:
Burst Time: 7
Arrival Time: 0
Process 4:
Burst Time: 6
Arrival Time: 0
Enter Time Quantum:     2
Process Burst Time      Arrival Time    Waiting Time    Turn Around Time
1               5               0               12              17
2               6               0               13              19
3               7               0               17              24
4               6               0               17              23
Average Waiting Time = 14.750000
Average Turnaround Time = 20.750000

--------------------------------
Process exited after 43.74 seconds with return value 0
Press any key to continue . . .
```

CSE 302L: Operating Systems Lab

LAB ASSESSMENT RUBRICS

Marking Criteria	**Exceeds expectation (2.5)**	**Meets expectation (1.5)**	**Does not meet expectation (0)**	**Score**
1. Correctness	Program compiles (no errors and no warnings). Program always works correctly and meets the specification(s). Completed between 81-100% of the requirements.	Program compiles (no errors and some warnings). Some details of the program specification are violated, program functions incorrectly for some inputs. Completed between 41-80% of the requirements.	Program fails to or compile with lots of warnings. Program only functions correctly in very limited cases or not at all. Completed less than 40% of the requirements.	
2. Delivery	Delivered on time, and in correct format (disk, email, hard copy etc.)	Not delivered on time, or slightly incorrect format.	Not delivered on time or not in correct format.	
3. Coding Standards	Proper indentation, whitespace, line length, wrapping, comments and references.	Missing some of whitespace, line length, wrapping, comments or references.	Poor use of whitespace, line length, wrapping, comments and references.	
4. Presentation of document	Includes name, date, and assignment title. Task titles, objectives, output screenshots included and good formatting and excellently organized.	Includes name, date, and assignment title. Task titles, objectives, output screenshots included and good formatting.	No name, date, or assignment title included. No task titles, no objectives, no output screenshots, poor formatting.	

Instructor:

Name: Engr. Abdullah Hamid

Signature: ________________________