Atma Ram Sanatan Dharma College

University of Delhi

Operating System

Practical File

Submitted By:

Jyotiswaroop Srivastav
College Roll No. 21/18023
Semester III
BSc. (Hons) Computer Science

Submitted To:

Ms. Parul Jain

9. Write a program to implement non-preemptive priority based scheduling algorithm.

```c
/**
 * Write a program to implement non-preemptive priority
 * based scheduling algorithm.
 */

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

struct process
{
    int pid;
    double priority;
    double burstTime;
    double arrivalTime;
    double waitingTime;
    double turnAroundTime;
};

int comparisonDesc(const void *a, const void *b)
{
    return ((struct process *)a)->priority < ((struct process
*)b)->priority;
}

int comparisonAsc(const void *a, const void *b)
{
    return ((struct process *)a)->pid > ((struct process *)b)-
>pid;
}

void computeWaitingTime(struct process *processes, int
processCount)
{
```

```
    qsort(processes, processCount, sizeof(struct process),
comparisonDesc);
    processes[0].waitingTime = 0;
    for (int i = 0; i < processCount - 1; i++)
        processes[i + 1].waitingTime =
            processes[i].burstTime +
            processes[i].waitingTime -
            processes[i].arrivalTime;
    return;
}

void computeTurnAroundTime(struct process *processes, int
processCount)
{
    for (int i = 0; i < processCount; i++)
        processes[i].turnAroundTime =
            processes[i].burstTime +
            processes[i].waitingTime -
            processes[i].arrivalTime;
    qsort(processes, processCount, sizeof(struct process),
comparisonAsc);
    return;
}

void printAverageTimes(struct process *processes, int
processCount, char *unit)
{
    double totalWaitingTime = 0.0;
    double totalTurnAroundTime = 0.0;
    computeWaitingTime(processes, processCount);
    computeTurnAroundTime(processes, processCount);
    printf("Process ID\tPriority\tBurst Time\tArrival
Time\tWaiting Time\tTurn-Around Time\n");
    printf("-----------------------------------------------
-----");
    printf("----------------------------------------\n");
    for (int i = 0; i < processCount; i++)
    {
```

```
        totalWaitingTime += processes[i].waitingTime;
        totalTurnAroundTime += processes[i].turnAroundTime;
        printf("%d\t\t%.2lf\t\t%.2lf%s\t\t%.2lf%s\t\t%.2lf%s\t\
t%.2lf%s\n",
               processes[i].pid, processes[i].priority,
               processes[i].burstTime, unit,
processes[i].arrivalTime, unit,
               processes[i].waitingTime, unit,
processes[i].turnAroundTime, unit);
    }
    printf("\nAverage Waiting Time = %.2lf%s", totalWaitingTime
/ processCount,
           unit);
    printf("\nAverage Turn-Around time = %.2lf%s\n",
           totalTurnAroundTime / processCount, unit);
    return;
}

int main(void)
{
    int processCount;
    char unit[4] = {'\0'};

    printf("Enter Time Unit: ");
    fgets(unit, 3, stdin);

    printf("Enter Number of Processes: ");
    scanf("%i", &processCount);

    struct process processes[processCount];

    for (int i = 0; i < processCount; i++)
    {
        processes[i].pid = i + 1;
        printf("Burst Time for Process %i: ", i + 1);
        scanf("%lf", &processes[i].burstTime);
        printf("Arrival Time for Process %i: ", i + 1);
        scanf("%lf", &processes[i].arrivalTime);
```

```c
        printf("Priority for Process %i: ", i + 1);
        scanf("%lf", &processes[i].priority);
    }

    printf("\n");

    printAverageTimes(processes, processCount, unit);

    return 0;
}
```

```
$ gcc -o main main.c
$ ./main
Enter Number of Processes: 5
Burst Time for Process 1: 3
Arrival Time for Process 1: 0
Priority for Process 1: 3
Burst Time for Process 2: 5
Arrival Time for Process 2: 0
Priority for Process 2: 4
Burst Time for Process 3: 1
Arrival Time for Process 3: 0
Priority for Process 3: 1
Burst Time for Process 4: 7
Arrival Time for Process 4: 0
Priority for Process 4: 7
Burst Time for Process 5: 4
Arrival Time for Process 5: 0
Priority for Process 5: 8

Process ID      Priority        Burst Time      Arrival Time    Waiting Time    Turn-Around Time
-------------------------------------------------------------------------------------------
1               3               3               0               16              19
2               4               5               0               11              16
3               1               1               0               19              20
4               7               7               0               4               11
5               8               4               0               0               4

Average Waiting Time = 10.00
Average Turn-Around time = 14.00
```

10. Write a program to implement preemptive priority based scheduling algorithm.

```c
/**
 * Write a program to implement preemptive priority
 * based scheduling algorithm.
 */

#include <stdio.h>
struct priority_scheduling
{
    char process_name;
    int burst_time;
    int waiting_time;
    int turn_around_time;
    int priority;
};

int main()
{
    int number_of_process;
    int total = 0;
    struct priority_scheduling temp_process;
    int ASCII_number = 65;
    int position;
    float average_waiting_time;
    float average_turnaround_time;

    printf("Enter the total number of Processes: ");
    scanf("%d", &number_of_process);

    // initializing the structure array
    struct priority_scheduling process[number_of_process];
    printf("\nPlease Enter the  Burst Time and Priority of each
process:\n");

    // get burst time and priority of all process
```

```
    for (int i = 0; i < number_of_process; i++)
    {
        // assign names consecutively using ASCII number
        process[i].process_name = (char)ASCII_number;

        printf("\nEnter the details of the process %c \n",
process[i].process_name);
        printf("Enter the burst time: ");
        scanf("%d", &process[i].burst_time);

        printf("Enter the priority: ");
        scanf("%d", &process[i].priority);

        ASCII_number++;
    }

    // swap process according to high priority
    for (int i = 0; i < number_of_process; i++)
    {

        position = I;

        for (int j = i + 1; j < number_of_process; j++)
        {

            // check if priority is higher for swapping
            if (process[j].priority >
process[position].priority)
                position = j;
        }
        // swapping of lower priority process with the higher
priority process
        temp_process = process[i];
        process[i] = process[position];
        process[position] = temp_process;
    }
    // First process will not have to wait and hence has a
waiting time of 0
```

```
    process[0].waiting_time = 0;

    for (int i = 1; i < number_of_process; i++)
    {
        process[i].waiting_time = 0;
        for (int j = 0; j < i; j++)
        {
            process[i].waiting_time += process[j].burst_time;
        }
        total += process[i].waiting_time;
    }

    average_waiting_time = (float)total /
(float)number_of_process;
    total = 0;

    printf("\n\nProcess_name \t Burst Time \t Waiting Time
\t  Turnaround Time\n");
    printf("-----------------------------------------------
---------\n");

    for (int i = 0; i < number_of_process; i++)
    {
        process[i].turn_around_time = process[i].burst_time +
process[i].waiting_time;
        total += process[i].turn_around_time;
        printf("\t  %c \t\t  %d \t\t %d \t\t %d",
process[i].process_name, process[i].burst_time,
process[i].waiting_time, process[i].turn_around_time);
        printf("\n-----------------------------------------
-------------\n");
    }

    // calculating the average turn_around time
    average_turnaround_time = (float)total /
(float)number_of_process;
```

```
    printf("\n\n Average Waiting Time : %f",
average_waiting_time);
    printf("\n Average Turnaround Time: %f\n",
average_turnaround_time);

    return 0;
}
```

```
Enter the details of the process A
Enter the burst time: 5
Enter the priority: 2
Enter the details of the process B
Enter the burst time: 6
Enter the priority: 1
Enter the details of the process C
Enter the burst time: 7
Enter the priority: 3

Process_name      Burst Time      Waiting Time      Turnaround Time
-----------------------------------------------------------
    C                 7              0                 7
-----------------------------------------------------------
    A                 5              7                 12
-----------------------------------------------------------
    B                 6              12                 18
-----------------------------------------------------------


Average Waiting Time: 6.333333
Average Turnaround Time: 12.333333
```