

# ST. JOSEPH'S COLLEGE OF ENGINEERING AND TECHNOLOGY Department of Artificial Intelligence and Data Science CSL204 OPERATING SYSTEMS LAB MANUAL

### **EXPERIMENT NO:6 C**

### SCHEDULING ALGORITHMS

### **AIM**

To implement different scheduling algorithms.

## C. Priority Scheduling

## **Algorithm**

- 1. Input the number of processes
  - $\circ$  Prompt the user to enter the number of processes n.
- 2. Input Burst Time and Priority for each process
  - o For each process, input its Burst Time (BT) and Priority (PR).
  - o Store the process IDs, burst times, and priorities in separate arrays.
- 3. Sort the processes based on priority (Ascending Order)
  - Use a sorting algorithm (like **Selection Sort**) to arrange processes such that the process with the highest priority (lowest priority number) comes first.
  - o If two processes have the same priority, they are executed in the order they were entered.
- 4. Calculate Waiting Time (WT)
  - The first process has WT = 0.
  - o For every other process: WT[i]=WT[i-1]+BT[i-1]
  - Accumulate total waiting time (tot\_wt).
- 5. Calculate Turnaround Time (TAT)
  - o For each process: TAT[i]=WT[i]+BT[i]
  - Accumulate total turnaround time (tot\_tat).
- 6. Compute Averages
  - o Compute Average Waiting Time:
    - Avg\_WT=Total\_WT/n
  - o Compute Average Turnaround Time:
    - Avg TAT=Total TAT/n
- 7. Display Process Details
  - o Print process IDs, burst times, priorities, waiting times, and turnaround times in a table format.
- 8. Print the Final Results
  - o Print Total Waiting Time, Total Turnaround Time, Average Waiting Time, and Average Turnaround Time.

## **PROGRAM**

#include <stdio.h> // Standard Input-Output library
// Function to swap two integer values



## ST. JOSEPH'S COLLEGE OF ENGINEERING AND TECHNOLOGY Department of Artificial Intelligence and Data Science CSL204 OPERATING SYSTEMS LAB MANUAL

```
void swap(int *a, int *b) {
  int temp = *a;
  *a = *b:
  *b = temp;
int main() {
                              // Declare necessary variables
  int i, j, n;
  int p[30], bt[30], pr[30], tat[30], wt[30];
  int tot_tat = 0, tot_wt = 0;
  float avg_tat, avg_wt;
                              // Prompt user to enter the number of processes
  printf("\nEnter the number of processes: ");
  scanf("%d", &n);
                              // Loop to take input for burst time and priority for each process
  for (i = 0; i < n; i++)
     printf("Enter burst time and priority of process[%d]: ", i + 1);
     scanf("%d%d", &bt[i], &pr[i]); // Read burst time and priority
    p[i] = i + 1; // Assign process number (1-based indexing for display)
                              // Sorting processes based on priority (lower value = higher priority)
  for (i = 0; i < n - 1; i++)
     for (j = i + 1; j < n; j++) {
       if (pr[i] > pr[j])
                              // Swap if priority of current is higher
{
          swap(&pr[i], &pr[j]);
          swap(&bt[i], &bt[i]);
          swap(&p[i], &p[j]);
       }
     }
  }
                              // Calculate Waiting Time (WT) and Turnaround Time (TAT)
                              // First process has 0 waiting time
  wt[0] = 0;
  tat[0] = bt[0];
                              // First process TAT = its burst time
  tot tat = tat[0];
  for (i = 1; i < n; i++)
     wt[i] = wt[i-1] + bt[i-1]; // Waiting Time = previous WT + previous BT
     tat[i] = wt[i] + bt[i];
                             // Turnaround Time = Waiting Time + Burst Time
     tot_wt += wt[i];
                              // Sum total waiting time
     tot_tat += tat[i];
                              // Sum total turnaround time
```



## ST. JOSEPH'S COLLEGE OF ENGINEERING AND TECHNOLOGY Department of Artificial Intelligence and Data Science CSL204 OPERATING SYSTEMS LAB MANUAL

```
// Compute average values

avg_wt = (float)tot_wt / n;

avg_tat = (float)tot_tat / n;

// Print Process Details

printf("\nPROCESS\t\tBURST TIME\tPRIORITY\tWAITING TIME\tTURNAROUND TIME");

for (i = 0; i < n; i++) {
    printf("\nProcess[%d]\t\t%d\t\t%d\t\t%d\t\t%d", p[i], bt[i], pr[i], wt[i], tat[i]);
}

// Print total and average turnaround and waiting times

printf("\n\nTotal Turnaround Time: %d", tot_tat);

printf("\nAverage Turnaround Time: %.2f", avg_tat);

printf("\nTotal Waiting Time: %d", tot_wt);

printf("\nAverage Waiting Time: %.2f\n", avg_wt);

return 0;
```

## **OUTPUT**

Enter the no.of processes 4

Enter burst time and priority of process[1]:5 2

Enter burst time and priority of process[2]:4 3

Enter burst time and priority of process[3]:6 1

Enter burst time and priority of process[4]:3 4

PROCESS	BURST TIMEPRIORITY		TURN AROUND TIME	WAITINGTIME
process[3]	6	1	6	0
process[1]	5	2	11	6
process[2]	4	3	15	11
process[4]	3	4	18	15

Total Turn around Time:50

Average Turn around Time:12

Total Waiting Time:32 Total avg. Waiting Time:8