ASSIGNMENT 5

**Aim:** To create C programs for the different scheduling algorithms.

To perform: Create and execute C programs for following CPU Scheduling Algorithms:

**🔹 1. First Come First Serve (FCFS)**

**✅ C Code:**

#include <stdio.h>

int main() {

int n, i;

printf("Enter number of processes: ");

scanf("%d", &n);

int bt[n], wt[n], tat[n];

wt[0] = 0;

printf("Enter burst time for each process:\n");

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

printf("P%d: ", i+1);

scanf("%d", &bt[i]);

}

// Calculate waiting time

for(i = 1; i < n; i++)

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

// Calculate turnaround time

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

tat[i] = wt[i] + bt[i];

printf("\nProcess\tBT\tWT\tTAT\n");

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

printf("P%d\t%d\t%d\t%d\n", i+1, bt[i], wt[i], tat[i]);

return 0;

}

**🖥️ Sample Output:**

yaml

CopyEdit

Enter number of processes: 3

Enter burst time for each process:

P1: 5

P2: 8

P3: 6

Process BT WT TAT

P1 5 0 5

P2 8 5 13

P3 6 13 19

**🔹 2. Shortest Job First (Non-Preemptive)**

**✅ C Code:**

c

#include <stdio.h>

void swap(int \*a, int \*b) {

int t = \*a;

\*a = \*b;

\*b = t;

}

int main() {

int n, i, j;

printf("Enter number of processes: ");

scanf("%d", &n);

int pid[n], bt[n], wt[n], tat[n];

printf("Enter burst time for each process:\n");

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

pid[i] = i + 1;

printf("P%d: ", pid[i]);

scanf("%d", &bt[i]);

}

// Sort by burst time

for(i = 0; i < n - 1; i++) {

for(j = i + 1; j < n; j++) {

if(bt[i] > bt[j]) {

swap(&bt[i], &bt[j]);

swap(&pid[i], &pid[j]);

}

}

}

wt[0] = 0;

for(i = 1; i < n; i++)

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

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

tat[i] = wt[i] + bt[i];

printf("\nProcess\tBT\tWT\tTAT\n");

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

printf("P%d\t%d\t%d\t%d\n", pid[i], bt[i], wt[i], tat[i]);

return 0;

}

**🖥️ Sample Output:**

yaml

CopyEdit

Enter number of processes: 3

Enter burst time for each process:

P1: 6

P2: 2

P3: 8

Process BT WT TAT

P2 2 0 2

P1 6 2 8

P3 8 8 16

**🔹 3. Round Robin Scheduling**

**✅ C Code:**

c

#include <stdio.h>

int main() {

int n, i, time = 0, tq;

printf("Enter number of processes: ");

scanf("%d", &n);

int bt[n], rt[n], wt[n], tat[n];

printf("Enter burst time for each process:\n");

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

printf("P%d: ", i+1);

scanf("%d", &bt[i]);

rt[i] = bt[i];

wt[i] = 0;

}

printf("Enter time quantum: ");

scanf("%d", &tq);

int done;

do {

done = 1;

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

if(rt[i] > 0) {

done = 0;

if(rt[i] > tq) {

time += tq;

rt[i] -= tq;

} else {

time += rt[i];

wt[i] = time - bt[i];

rt[i] = 0;

}

}

}

} while(!done);

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

tat[i] = bt[i] + wt[i];

printf("\nProcess\tBT\tWT\tTAT\n");

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

printf("P%d\t%d\t%d\t%d\n", i+1, bt[i], wt[i], tat[i]);

return 0;

}

**🖥️ Sample Output:**

yaml

CopyEdit

Enter number of processes: 3

Enter burst time for each process:

P1: 10

P2: 5

P3: 8

Enter time quantum: 3

Process BT WT TAT

P1 10 13 23

P2 5 9 14

P3 8 14 22