**LAB ASSIGNMENT 5**

**1.FIRST COME FIRST SERVE(FCFS)**

#include <stdio.h>

int main() {

int n, i;

printf("Enter number of processes: ");

scanf("%d", &n);

int bt[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]);

}

wt[0] = 0;

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

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

}

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

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

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

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

}

return 0;

}

**OUTPUT:**

Enter number of processes: 4

Enter burst time for each process:

P1: 10

P2: 12

P3: 5

P4: 7

Process BT WT TAT

P1 10 0 10

P2 12 10 22

P3 5 22 27

P4 7 27 34

**2.SHORTEST JOB FIRST(SJF) NON PREEMPTIVE**

#include <stdio.h>

int main() {

int n, i, j;

printf("Enter number of processes: ");

scanf("%d", &n);

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

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

p[i] = i+1;

printf("Enter burst time for P%d: ", i+1);

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

}

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

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

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

temp = bt[i]; bt[i] = bt[j]; bt[j] = temp;

temp = p[i]; p[i] = p[j]; p[j] = temp;

}

}

}

wt[0] = 0;

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

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

}

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

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

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

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

}

return 0;

}

**OUTPUT:**

Enter number of processes: 4

Enter burst time for P1: 10

Enter burst time for P2: 12

Enter burst time for P3: 5

Enter burst time for P4: 7

Process BT WT TAT

P3 5 0 5

P4 7 5 12

P1 10 12 22

P2 12 22 34

**ROUND ROBIN SCHEDULING:**

#include <stdio.h>

int main() {

int n, tq, i, time = 0;

printf("Enter number of processes: ");

scanf("%d", &n);

int bt[n], rem\_bt[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]);

rem\_bt[i] = bt[i];

}

printf("Enter time quantum: ");

scanf("%d", &tq);

while(1) {

int done = 1;

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

if(rem\_bt[i] > 0) {

done = 0;

if(rem\_bt[i] > tq) {

time += tq;

rem\_bt[i] -= tq;

} else {

time += rem\_bt[i];

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

rem\_bt[i] = 0;

}

}

}

if(done) break;

}

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

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

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

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

}

return 0;

}

**OUTPUT:**

Enter number of processes: 4

Enter burst time for each process:

P1: 10

P2: 15

P3: 5

P4: 7

Enter time quantum: 7

Process BT WT TAT

P1 10 19 29

P2 15 22 37

P3 5 14 19

P4 7 19 26