# PRACTICAL – 6

**Aim:** **Write a C program in LINUX to implement Process scheduling algorithms and compare.**

**A**: **First Come First Serve (FCFS) Scheduling**

**PROGRAM:**

#include <stdio.h>

int waitingtime(int proc[], int n, int burst\_time[], int wait\_time[])

{

wait\_time[0] = 0;

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

wait\_time[i] = burst\_time[i - 1] + wait\_time[i - 1];

return 0;

}

int turnaroundtime(int proc[], int n, int burst\_time[], int wait\_time[], int tat[])

{

int i;

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

tat[i] = burst\_time[i] + wait\_time[i];

return 0;

}

int avgtime(int proc[], int n, int burst\_time[])

{

int wait\_time[n], tat[n], total\_wt = 0, total\_tat = 0;

int i;

waitingtime(proc, n, burst\_time, wait\_time);

turnaroundtime(proc, n, burst\_time, wait\_time, tat);

printf("Processes Burst Waiting Turn around \n");

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

{

total\_wt = total\_wt + wait\_time[i];

total\_tat = total\_tat + tat[i];

printf(" %d\t %d\t\t %d \t%d\n", i + 1, burst\_time[i], wait\_time[i], tat[i]);

}

printf("Average waiting time = %f\n", (float)total\_wt / (float)n);

printf("Average turn around time = %f\n", (float)total\_tat / (float)n);

return 0;

}

int main()

{

int proc[] = {1, 2, 3};

int n = sizeof proc / sizeof proc[0];

int burst\_time[] = {4, 7, 10};

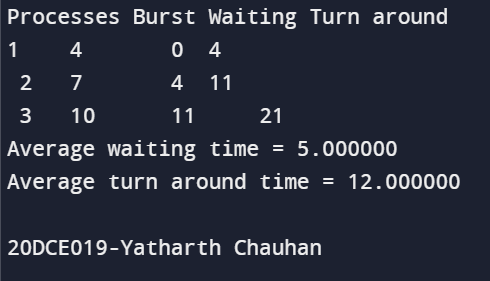
avgtime(proc, n, burst\_time);

printf("\n20DCE019-Yatharth Chauhan\n");

return 0;

}

**OUTPUT:**

****

**B**: **Write a program to execute fork() and find out the process id by getpid() system call.**

**PROGRAM:**

#include <stdio.h>

int main()

{

int A[100][4];

int i, j, n, total = 0, index, temp;

float avg\_wt, avg\_tat;

printf("Enter number of process: ");

scanf("%d", &n);

printf("Enter Burst Time:\n");

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

{

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

scanf("%d", &A[i][1]);

A[i][0] = i + 1;

}

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

{

index = i;

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

if (A[j][1] < A[index][1])

index = j;

temp = A[i][1];

A[i][1] = A[index][1];

A[index][1] = temp;

temp = A[i][0];

A[i][0] = A[index][0];

A[index][0] = temp;

}

A[0][2] = 0;

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

{

A[i][2] = 0;

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

A[i][2] += A[j][1];

total += A[i][2];

}

avg\_wt = (float)total / n;

total = 0;

printf("P BT WT TAT\n");

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

{

A[i][3] = A[i][1] + A[i][2];

total += A[i][3];

printf("P%d %d %d %d\n", A[i][0],

A[i][1], A[i][2], A[i][3]);

}

avg\_tat = (float)total / n;

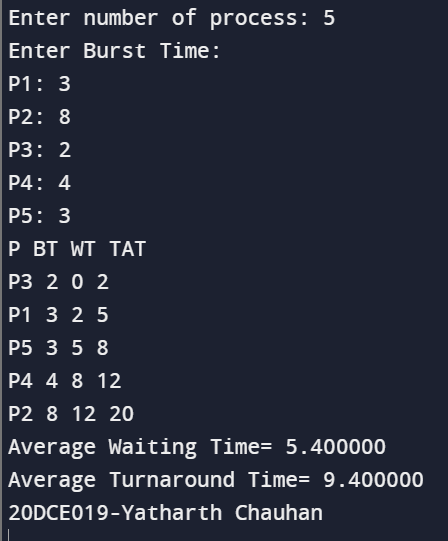
printf("Average Waiting Time= %f", avg\_wt);

printf("\nAverage Turnaround Time= %f", avg\_tat);

printf("\n20DCE019-Yatharth Chauhan\n");

}

**OUTPUT:**

****

**C**: **Write a program to execute following system call fork(), execl(), getpid(), exit(), wait() for a process.**

**PROGRAM:**

#include <stdio.h>

int main()

{

int bt[20], p[20], wt[20], tat[20], pr[20], i, j, n, total = 0, pos, temp, avg\_wt, avg\_tat;

printf("Enter Total Number of Process:");

scanf("%d", &n);

printf("\nEnter Burst Time and Priority\n");

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

{

printf("\nP[%d]\n", i + 1);

printf("Burst Time:");

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

printf("Priority:");

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

p[i] = i + 1;

}

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

{

pos = i;

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

{

if (pr[j] < pr[pos])

pos = j;

}

temp = pr[i];

pr[i] = pr[pos];

pr[pos] = temp;

temp = bt[i];

bt[i] = bt[pos];

bt[pos] = temp;

temp = p[i];

p[i] = p[pos];

p[pos] = temp;

}

wt[0] = 0;

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

{

wt[i] = 0;

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

wt[i] += bt[j];

total += wt[i];

}

avg\_wt = total / n;

total = 0;

printf("\nProcess\t Burst Time \tWaiting Time\tTurnaround Time");

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

{

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

total += tat[i];

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

}

avg\_tat = total / n;

printf("\n\nAverage Waiting Time=%d", avg\_wt);

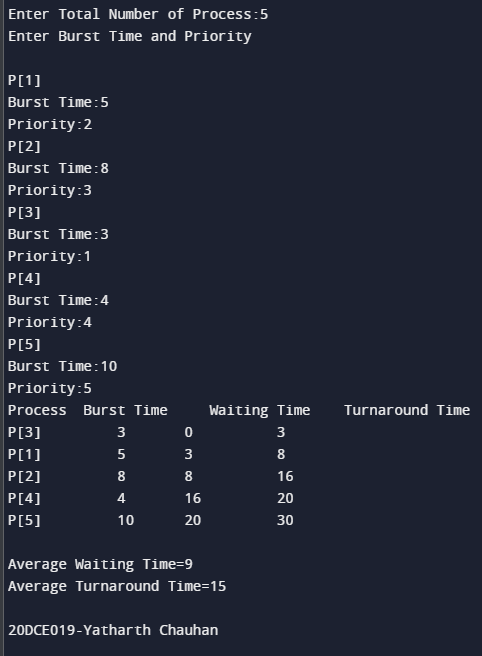
printf("\nAverage Turnaround Time=%d\n", avg\_tat);

printf("\n20DCE019-Yatharth Chauhan\n");

return 0;

}

**OUTPUT:**

****

**D**: **Write a program to find out status of named file (program of working stat() system call.**

**PROGRAM:**

#include <stdio.h>

int main()

{

int i, limit, total = 0, x, counter = 0, time\_quantum;

int wait\_time = 0, turnaround\_time = 0, arrival\_time[10], burst\_time[10], temp[10];

float average\_wait\_time, average\_turnaround\_time;

printf("\nEnter Total Number of Processes: ");

scanf("%d", &limit);

x = limit;

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

{

printf("\nEnter Details of Process[%d]\n", i + 1);

printf("Arrival Time:\t");

scanf("%d", &arrival\_time[i]);

printf("Burst Time:\t");

scanf("%d", &burst\_time[i]);

temp[i] = burst\_time[i];

}

printf("\nEnter Time Quantum:\t");

scanf("%d", &time\_quantum);

printf("\nProcess IDttBurst Timet Turnaround Timet Waiting Timen");

for (total = 0, i = 0; x != 0;)

{

if (temp[i] <= time\_quantum && temp[i] > 0)

{

total = total + temp[i];

temp[i] = 0;

counter = 1;

}

else if (temp[i] > 0)

{

temp[i] = temp[i] - time\_quantum;

total = total + time\_quantum;

}

if (temp[i] == 0 && counter == 1)

{

x--;

printf("\nProcess[%d]\t%d\t %d\t %d", i + 1, burst\_time[i], total - arrival\_time[i],

total - arrival\_time[i] - burst\_time[i]);

wait\_time = wait\_time + total - arrival\_time[i] - burst\_time[i];

turnaround\_time = turnaround\_time + total - arrival\_time[i];

counter = 0;

}

if (i == limit - 1)

{

i = 0;

}

else if (arrival\_time[i + 1] <= total)

{

i++;

}

else

{

i = 0;

}

}

average\_wait\_time = wait\_time \* 1.0 / limit;

average\_turnaround\_time = turnaround\_time \* 1.0 / limit;

printf("\nnAverage Waiting Time:\t%f", average\_wait\_time);

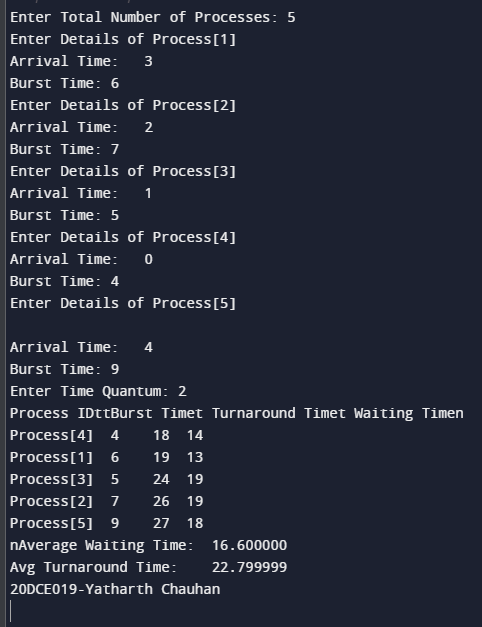
printf("\nAvg Turnaround Time:\t%f", average\_turnaround\_time);

printf("\n20DCE019-Yatharth Chauhan\n");

return 0;

}

**OUTPUT:**

****