ROUND ROBIN

#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;

int choice, num;

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

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];

}

while(1)

{

printf("1. SRTF \n");

printf("2. RR\n");

printf("4. Exit\n\n\n");

printf("Enter your choice : ");

scanf("%d",&choice);

switch(choice)

{

case 1:

printf("\n\nSRTF");

break;

case 2:

printf("nEnter Time Quantum:t");

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

printf("nProcess ID Burst Time Turnaround Time Waiting Time /n");

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] \n Burst time %d\n Turn around time %d\n Waitig time%d\n", 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%fn", average\_turnaround\_time);

return 0;

case 4:

printf("\n\n\t\t\tCoding is Fun !\n\n\n");

exit(0); // terminates the complete program execution

}

}

}

SRTF

#include <stdio.h>

int main()

{

int arrival\_time[10], burst\_time[10], temp[10];

int i, smallest, count = 0, time, limit;

double wait\_time = 0, turnaround\_time = 0, end;

float average\_waiting\_time, average\_turnaround\_time;

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

scanf("%d", &limit);

printf("\nEnter Details of %d Processes\n", limit);

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

{

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

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

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

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

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

}

burst\_time[9] = 9999;

for(time = 0; count != limit; time++)

{

smallest = 9;

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

{

if(arrival\_time[i] <= time && burst\_time[i] < burst\_time[smallest] && burst\_time[i] > 0)

{

smallest = i;

}

}

burst\_time[smallest]--;

if(burst\_time[smallest] == 0)

{

count++;

end = time + 1;

wait\_time = wait\_time + end - arrival\_time[smallest] - temp[smallest];

turnaround\_time = turnaround\_time + end - arrival\_time[smallest];

}

}

average\_waiting\_time = wait\_time / limit;

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

printf("\n\nAverage Waiting Time:\t%lf\n", average\_waiting\_time);

printf("Average Turnaround Time:\t%lf\n", average\_turnaround\_time);

return 0;

}