WITH ARRIVAL TIME --

#include <stdio.h>

void priority\_preemptive(int processes[], int n, int burst\_time[], int priority[], int arrival\_time[])

{

int remaining\_time[n];

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

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

int completed = 0, current\_time = 0;

while (completed != n)

{

int highest\_priority = -1;

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

{

if (arrival\_time[i] <= current\_time && remaining\_time[i] > 0)

{

if (highest\_priority == -1 || priority[i] < priority[highest\_priority])

highest\_priority = i;

}

}

if (highest\_priority == -1)

{

current\_time++;

continue;

}

printf("Executing process %d at time %d\n", processes[highest\_priority], current\_time);

remaining\_time[highest\_priority]--;

current\_time++;

if (remaining\_time[highest\_priority] == 0)

{

completed++;

printf("Process %d completed at time %d\n", processes[highest\_priority], current\_time);

}

}

}

int main()

{

int n;

printf("Enter the number of processes: ");

scanf("%d", &n);

int processes[n];

int burst\_time[n];

int priority[n];

int arrival\_time[n];

printf("Enter the burst time, priority, and arrival time for each process:\n");

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

{

printf("Process %d:\n", i + 1);

printf("Burst Time: ");

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

printf("Priority: ");

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

printf("Arrival Time: ");

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

processes[i] = i + 1;

}

priority\_preemptive(processes, n, burst\_time, priority, arrival\_time);

return 0;

}

WITHOUT ARRIVAL TIME

-->

#include <stdio.h>

void priority\_preemptive(int processes[], int n, int burst\_time[], int priority[])

{

int remaining\_time[n];

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

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

int completed = 0, current\_time = 0;

while (completed != n)

{

int highest\_priority = -1;

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

{

if (remaining\_time[i] > 0)

{

if (highest\_priority == -1 || priority[i] < priority[highest\_priority])

highest\_priority = i;

}

}

printf("Executing process %d at time %d\n", processes[highest\_priority], current\_time);

remaining\_time[highest\_priority]--;

current\_time++;

if (remaining\_time[highest\_priority] == 0)

{

completed++;

printf("Process %d completed at time %d\n", processes[highest\_priority], current\_time);

}

}

}

int main()

{

int n;

printf("Enter the number of processes: ");

scanf("%d", &n);

int processes[n];

int burst\_time[n];

int priority[n];

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

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

{

printf("Process %d:\n", i + 1);

printf("Burst Time: ");

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

printf("Priority: ");

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

processes[i] = i + 1;

}

priority\_preemptive(processes, n, burst\_time, priority);

return 0;

}