***//Assignment-10 Operating Systems Lab***

***//A program to implement preemptive priority based scheduling algorithm***

#include<stdio.h>

struct process

{

char process\_name;

int arrival\_time, burst\_time, ct, waiting\_time, turnaround\_time, priority;

int status;

}process\_queue[10];

int limit;

void Arrival\_Time\_Sorting()

{

struct process temp;

int i, j;

for(i = 0; i < limit - 1; i++)

{

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

{

if(process\_queue[i].arrival\_time > process\_queue[j].arrival\_time)

{

temp = process\_queue[i];

process\_queue[i] = process\_queue[j];

process\_queue[j] = temp;

}

}

}

}

int main()

{

int i, time = 0, burst\_time = 0, largest;

char c;

float wait\_time = 0, turnaround\_time = 0, average\_waiting\_time, average\_turnaround\_time;

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

scanf("%d", &limit);

for(i = 0, c = 'A'; i < limit; i++, c++)

{

process\_queue[i].process\_name = c;

printf("\nEnter Details For Process[%C]:\n", process\_queue[i].process\_name);

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

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

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

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

printf("Enter Priority:\t");

scanf("%d", &process\_queue[i].priority);

process\_queue[i].status = 0;

burst\_time = burst\_time + process\_queue[i].burst\_time;

}

Arrival\_Time\_Sorting();

process\_queue[9].priority = -9999;

printf("\nProcess Name\tArrival Time\tBurst Time\tPriority\tWaiting Time");

for(time = process\_queue[0].arrival\_time; time < burst\_time;)

{

largest = 9;

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

{

if(process\_queue[i].arrival\_time <= time && process\_queue[i].status != 1 && process\_queue[i].priority > process\_queue[largest].priority)

{

largest = i;

}

}

time = time + process\_queue[largest].burst\_time;

process\_queue[largest].ct = time;

process\_queue[largest].waiting\_time = process\_queue[largest].ct - process\_queue[largest].arrival\_time - process\_queue[largest].burst\_time;

process\_queue[largest].turnaround\_time = process\_queue[largest].ct - process\_queue[largest].arrival\_time;

process\_queue[largest].status = 1;

wait\_time = wait\_time + process\_queue[largest].waiting\_time;

turnaround\_time = turnaround\_time + process\_queue[largest].turnaround\_time;

printf("\n%c\t\t%d\t\t%d\t\t%d\t\t%d", process\_queue[largest].process\_name, process\_queue[largest].arrival\_time, process\_queue[largest].burst\_time, process\_queue[largest].priority, process\_queue[largest].waiting\_time);

}

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

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

printf("\n\nAverage waiting time:\t%f\n", average\_waiting\_time);

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

return 0;

}

**OUTPUT:**

Enter Total Number of Processes: 3

Enter Details For Process[A]:

Enter Arrival Time: 1

Enter Burst Time: 10

Enter Priority: 9

Enter Details For Process[B]:

Enter Arrival Time: 3

Enter Burst Time: 15

Enter Priority: 5

Enter Details For Process[C]:

Enter Arrival Time: 4

Enter Burst Time: 20

Enter Priority: 2

Process Name Arrival Time Burst Time Priority Waiting Time

A 1 10 9 0

B 3 15 5 8

C 4 20 2 22

Average waiting time: 10.000000

Average Turnaround Time: 25.000000