**Experiment No: 7**

|  |
| --- |
| **Student Name and Roll Number:** Namit Kumar |
| **Semester /Section:** V/FSA-1 |
| **Link to Code:** https://github.com/NamitKumar16/OS |
| **Date:** 6th October 2021 |
| **Faculty Signature:** |
| **Marks:** |

|  |
| --- |
| **Objective:**  Write a program to perform priority scheduling among a set of processes. |
| **Outcome:**  Student will understand the working of priority scheduling among a set of processes. |
| **Problem Statement:**  Implement the priority scheduling. |
| **Background Study:**  Priority scheduling is a non-preemptive algorithm and used in batch systems. Each process is assigned a priority. Process with highest priority is to be executed first and so on. |
| **Question Bank:**   1. What are advantages of Priority scheduling? 2. What are disadvantages of priority scheduling? 3. At the ready queue when a process arrives In [priority scheduling](https://t4tutorials.com/priority-based-process-scheduling-in-operating-systems/) algorithm, the priority of this process is compared with the priority of? A. currently running process B. parent process C. all process D. init process 4. Differentiate between pre-emptive and non pre-emptive scheduling? 5. What is total no. of queue required to perform Priority Scheduling ? |

**Student Work Area**

**Q1**

* Easy to use scheduling method
* Processes are executed on the basis of priority so high priority does not need to wait for long which saves time
* This method provides a good mechanism where the relative important of each process may be precisely defined.
* Suitable for applications with fluctuating time and resource requirements.

**Q2**

* If the system eventually crashes, all low priority processes get lost.
* If high priority processes take lots of CPU time, then the lower priority processes may starve and will be postponed for an indefinite time.
* This scheduling algorithm may leave some low priority processes waiting indefinitely.
* A process will be blocked when it is ready to run but has to wait for the CPU because some other process is running currently.
* If a new higher priority process keeps on coming in the ready queue, then the process which is in the waiting state may need to wait for a long duration of time.

**Q3** A – Currently Running Process

**Q4** – The basic difference between preemptive and non-preemptive scheduling is that in preemptive scheduling the CPU is allocated to the processes for the limited time. While in Non-preemptive scheduling, the CPU is allocated to the process till it terminates or switches to waiting state.

**Algorithm/Flowchart/Code/Sample Outputs**

#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] || (pr[j]==pr[pos] && bt[j]>bt[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);

return 0;

}

Text

Description automatically generated