Bharatiya Vidya Bhavan's

Sardar Patel Institute of Technology

Bhavan's Campus, Munshi Nagar, Andheri (West), Mumbai-400058-India
(Autonomous College Affiliated to University of Mumbai)

Computer Engineering Department & Information Technology Engineering Department

Academic Year: 2021-2022

Class: S.Y.B.Tech **Sem.:** 4 **Course:** OS

Name	**Pratik Pujari**		
UID no.	**2020300054**	**Class:**	**Comps C Batch**
Experiment No.	5		

AIM:	To implement the CPU scheduling algorithms

THEORY:

What is CPU Scheduling?

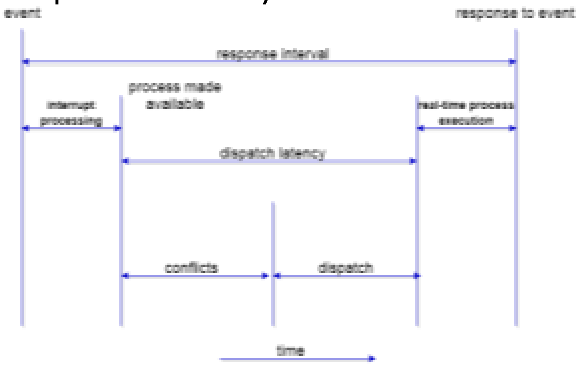
CPU scheduling is a process that allows one process to use the CPU while the execution of another process is on hold(in waiting state) due to unavailability of any resource like I/O etc, thereby making full use of CPU. The aim of CPU scheduling is to make the system efficient, fast, and fair.

CPU Scheduling: Dispatcher

Another component involved in the CPU scheduling function is the Dispatcher. The dispatcher is the module that gives control of the CPU to the process selected by the short-term scheduler. This function involves:

- Switching context
- Switching to user mode
- Jumping to the proper location in the user program to restart that program from where it left last time.

The dispatcher should be as fast as possible, given that it is invoked during every process switch. The time taken by the dispatcher to stop one process and start another process is known as the Dispatch Latency.



Computer Engineering Department &
Information Technology Engineering Department

Academic Year: 2021-2022

Class: S.Y.B.Tech **Sem.:** 4 **Course:** OS

CPU Scheduling: Scheduling Criteria

There are many different criteria to check when considering the "best" scheduling algorithm, they are:

CPU Utilization

To make out the best use of the CPU and not to waste any CPU cycle, the CPU would be working most of the time(Ideally 100% of the time). Considering a real system, CPU usage should range from 40% (lightly loaded) to 90% (heavily loaded.)

Throughput

It is the total number of processes completed per unit of time or rather says the total amount of work done in a unit of time. This may range from 10/second to 1/hour depending on the specific processes.

Turnaround Time

It is the amount of time taken to execute a particular process, i.e. The interval from the time of submission of the process to the time of completion of the process(Wall clock time).

Waiting Time

The sum of the periods spent waiting in the ready queue amount of time a process has been waiting in the ready queue to acquire get control on the CPU.

Load Average

It is the average number of processes residing in the ready queue waiting for their turn to get into the CPU.

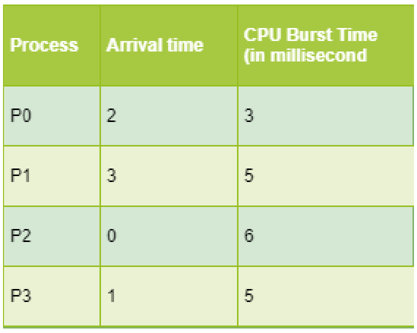
Response Time

Amount of time it takes from when a request was submitted until the first response is produced. Remember, it is the time till the first response and not the completion of process execution(final response).

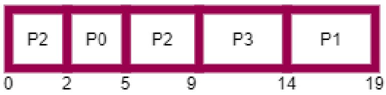
In general CPU utilization and Throughput are maximized and other factors are reduced for proper optimization.

Preemptive Scheduling

In this type of Scheduling, the tasks are usually assigned with priorities. At times it is necessary to run a certain task that has a higher priority before another task although it is running. Therefore, the running task is interrupted for some time and resumed later when the priority task has finished its execution. Some Algorithms that are based on preemptive scheduling are Round Robin Scheduling (RR), Shortest Remaining Time First (SRTF), Priority (preemptive version) Scheduling, etc.

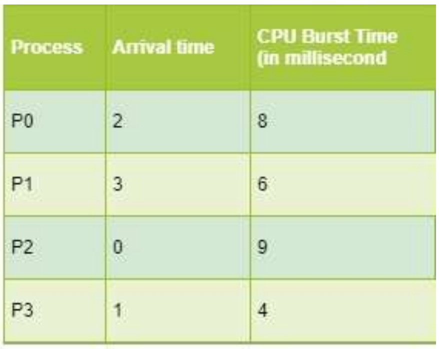


Process	Arrival time	CPU Burst Time (in millisecond
P0	2	3
P1	3	5
P2	0	6
P3	1	5

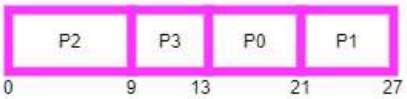


Non-Preemptive Scheduling

Under non-preemptive scheduling, once the CPU has been allocated to a process, the process keeps the CPU until it releases the CPU either by terminating or by switching to the waiting state.
This scheduling method is used by the Microsoft Windows 3.1 and by the Apple Macintosh operating systems.
It is the only method that can be used on certain hardware platforms because It does not require the special hardware(for example a timer) needed for preemptive scheduling.



Process	Arrival time	CPU Burst Time (in millisecond
P0	2	8
P1	3	6
P2	0	9
P3	1	4



Bharatiya Vidya Bhavan's

Sardar Patel Institute of Technology

Bhavan's Campus, Munshi Nagar, Andheri (West), Mumbai-400058-India
(Autonomous College Affiliated to University of Mumbai)

Computer Engineering Department & Information Technology Engineering Department

Academic Year: 2021-2022

Class: S.Y.B.Tech **Sem.:** 4 **Course:** OS

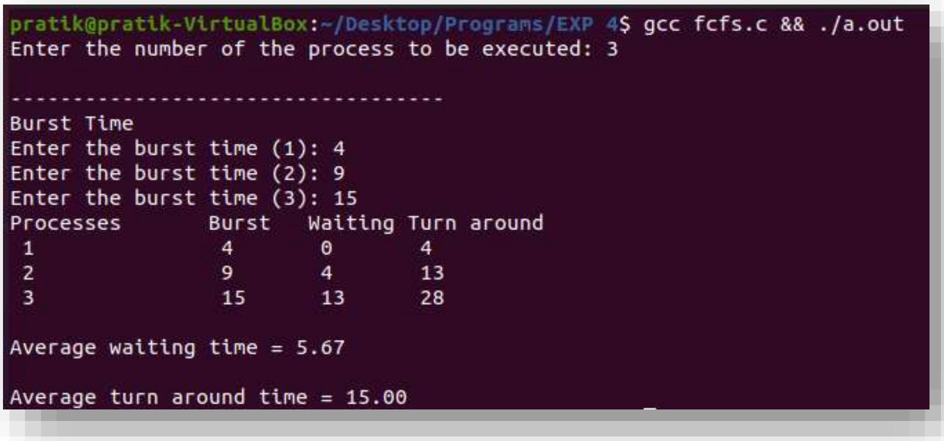
First Come First Serve Program

CODE

```
#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\tBurst\tWaiting\tTurn around \n");

   for ( i=0; i<n; i++) {
      total_wt = total_wt + wait_time[i];
```

```
     total_tat = total_tat + tat[i];
     printf(" %d\t\t %d\t %d\t %d\n", i+1, burst_time[i],
wait_time[i], tat[i]);
  }
  printf("\nAverage waiting time = %.2f\n", (float)total_wt /
(float)n);
  printf("\nAverage turn around time = %.2f\n",
(float)total_tat / (float)n);
  return 0;
}

int main() {

  printf("Enter the number of the process to be executed: ");
  int size;
  scanf("%d",&size);
  int proc[size] ;

  for(int i=0;i<size;i++){
      proc[i]=i+1;

  }

  int n = sizeof proc / sizeof proc[0];
  printf("\n----------------------------------");
  printf("\t\t\nBurst Time\n");
  int burst_time[size];
  for(int i=0;i<size;i++){
      printf("Enter the burst time (%d): ",(i+1));
      scanf("%d",&burst_time[i]);
  }
  avgtime(proc, n, burst_time);
  return 0;
}
```

OUTPUT:



```
pratik@pratik-VirtualBox:~/Desktop/Programs/EXP 4$ gcc fcfs.c && ./a.out
Enter the number of the process to be executed: 3

-----------------------------------
Burst Time
Enter the burst time (1): 4
Enter the burst time (2): 9
Enter the burst time (3): 15
Processes       Burst   Waiting Turn around
 1               4       0       4
 2               9       4       13
 3               15      13      28

Average waiting time = 5.67

Average turn around time = 15.00
```

Round Robin Program

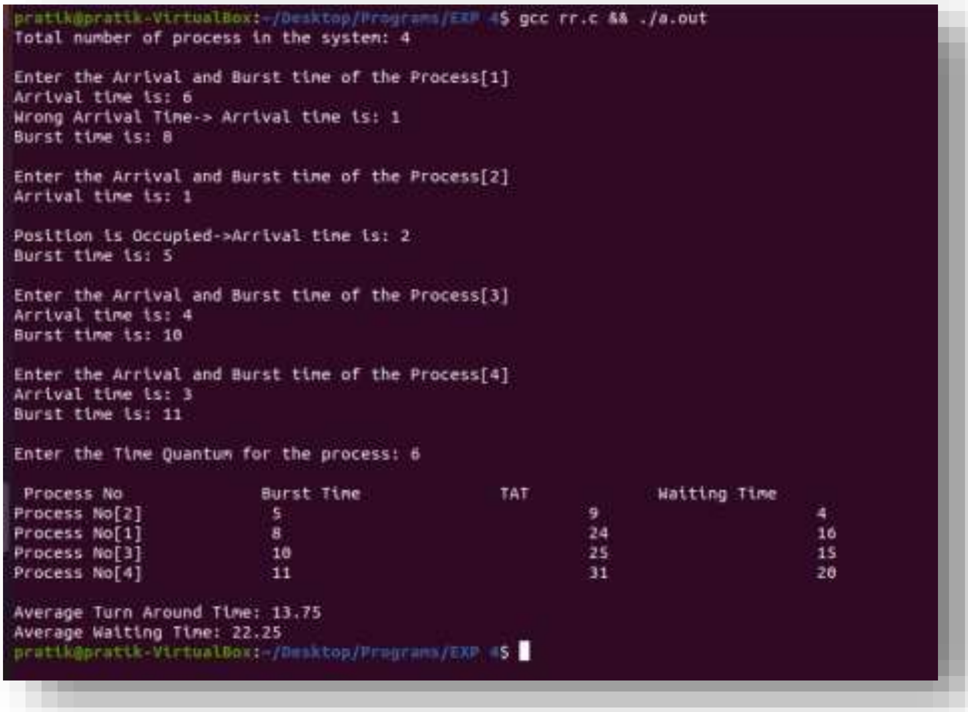
CODE:

```
#include<stdio.h>


void main()
{
    // initlialize the variable name
    int i, NOP, sum=0,count=0, y, quant, wt=0, tat=0, at[10],
bt[10], temp[10];
    float avg_wt, avg_tat;
    printf("Total number of process in the system: ");
    scanf("%d", &NOP);
    int position[NOP];
    y = NOP; // Assign the number of process to variable y

// Use for loop to enter the details of the process like Arrival
time and the Burst Time
        for(int i=0;i<NOP;i++)
        position[i]=0;
        for(i=0; i<NOP; i++)
        {
```

```
	printf("\nEnter the Arrival and Burst time of the Process[%d]\n", i+1);

	while(1){
		printf("Arrival time is: ");  // Accept arrival time
		int pos;
		scanf("%d", &pos);
		if(NOP<pos || pos<0){
			printf("Wrong Arrival Time-> ");
			continue;
		}

		else if(position[pos-1]==0){
			position[pos-1]=1;
			at[i]=pos;
			break;
		}
		else {
			printf("\nPosition is Occupied->");
		}

	}
	printf("Burst time is: "); // Accept the Burst time
	scanf("%d", &bt[i]);
	temp[i] = bt[i]; // store the burst time in temp array
	}
	// Accept the Time qunat
	printf("\nEnter the Time Quantum for the process: ");
	scanf("%d", &quant);
	// Display the process No, burst time, Turn Around Time and the waiting time
	printf("\n Process No \t\t Burst Time \t\t TAT \t\t Waiting Time ");
	for(sum=0, i = 0; y!=0; )
	{
	if(temp[i] <= quant && temp[i] > 0) // define the conditions
	{
```

```
           sum = sum + temp[i];
           temp[i] = 0;
           count=1;
           }
           else if(temp[i] > 0)
           {
              temp[i] = temp[i] - quant;
              sum = sum + quant;
           }
           if(temp[i]==0 && count==1)
           {
              y--; //decrement the process no.
              printf("\nProcess No[%d] \t\t  %d\t\t\t\t
%d\t\t\t %d", i+1, bt[i], sum-at[i], sum-at[i]-bt[i]);
              wt = wt+sum-at[i]-bt[i];
              tat = tat+sum-at[i];
              count =0;
           }
           if(i==NOP-1)
           {
              i=0;
           }
           else if(at[i+1]<=sum)
           {
              i++;
           }
           else
           {
              i=0;
           }
      }
      // represents the average waiting time and Turn Around
time
      avg_wt = wt * 1.0/NOP;
      avg_tat = tat * 1.0/NOP;
      printf("\n\nAverage Turn Around Time: %.2f", avg_wt);
      printf("\nAverage Waiting Time: %.2f\n", avg_tat);
}
```

OUTPUT:



```
pratik@pratik-VirtualBox:~/Desktop/Programs/EXP 4$ gcc rr.c && ./a.out
Total number of process in the system: 4

Enter the Arrival and Burst time of the Process[1]
Arrival time is: 6
Wrong Arrival Time-> Arrival time is: 1
Burst time is: 8

Enter the Arrival and Burst time of the Process[2]
Arrival time is: 1

Position is Occupied->Arrival time is: 2
Burst time is: 5

Enter the Arrival and Burst time of the Process[3]
Arrival time is: 4
Burst time is: 10

Enter the Arrival and Burst time of the Process[4]
Arrival time is: 3
Burst time is: 11

Enter the Time Quantum for the process: 6

 Process No              Burst Time              TAT             Waiting Time
Process No[2]             5                             9                       4
Process No[1]             8                             24                      16
Process No[3]             10                            25                      15
Process No[4]             11                            31                      20

Average Turn Around Time: 13.75
Average Waiting Time: 22.25
pratik@pratik-VirtualBox:~/Desktop/Programs/EXP 4$ 
```

Error Handling: User can input a incorrect arrival time

Bharatiya Vidya Bhavan's

Sardar Patel Institute of Technology

Bhavan's Campus, Munshi Nagar, Andheri (West), Mumbai-400058-India
(Autonomous College Affiliated to University of Mumbai)

Computer Engineering Department & Information Technology Engineering Department

Academic Year: 2021-2022

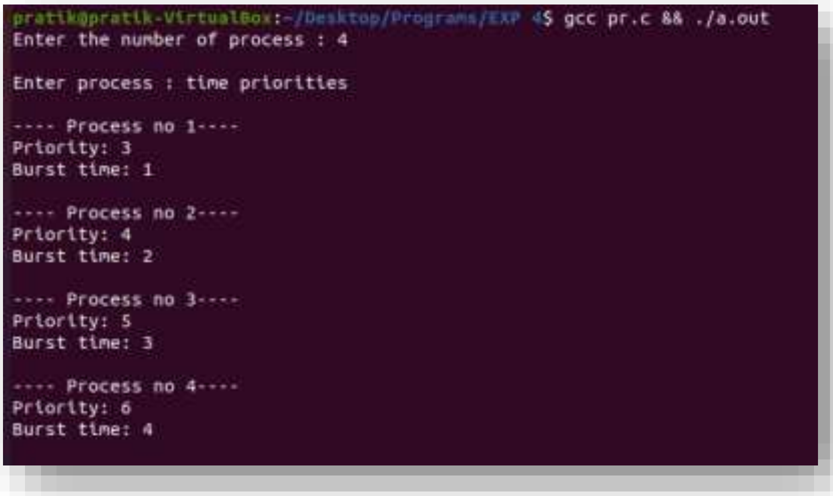
Class: S.Y.B.Tech **Sem.:** 4 **Course:** OS

Priority Scheduling Program

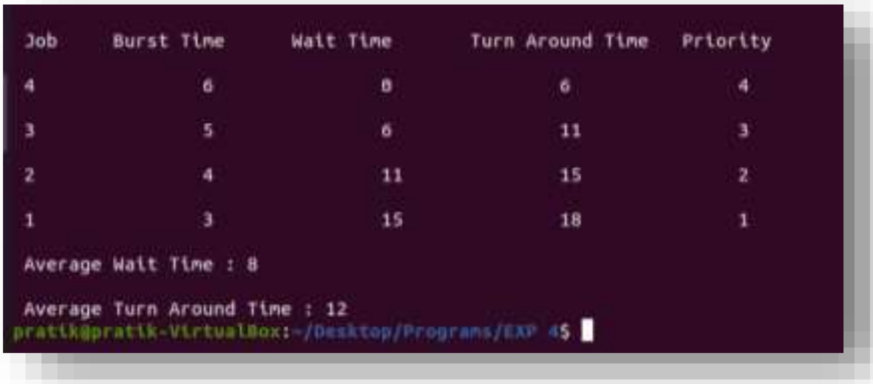
CODE:

```
#include<stdio.h>
void main()
 {
   int x,n,p[10],pp[10],pt[10],w[10],t[10],awt,atat,i;
   printf("Enter the number of process : ");
   scanf("%d",&n);
   printf("\nEnter process : time priorities \n");
   for(i=0;i<n;i++)
    {
      printf("\n---- Process no %d---- ",i+1);
      printf("\nPriority: ");
      scanf("%d",&pt[i]);
      printf("Burst time: ");
      scanf("%d",&pp[i]);
      p[i]=i+1;
    }
  for(i=0;i<n-1;i++)
   {
     for(int j=i+1;j<n;j++)
     {
       if(pp[i]<pp[j])
       {
         x=pp[i];
         pp[i]=pp[j];
         pp[j]=x;
         x=pt[i];
         pt[i]=pt[j];
         pt[j]=x;
         x=p[i];
         p[i]=p[j];
         p[j]=x;
      }
   }
        }
        w[0]=0;
        awt=0;
        t[0]=pt[0];
        atat=t[0];
        for(i=1;i<n;i++)
         {
           w[i]=t[i-1];
           awt+=w[i];
           t[i]=w[i]+pt[i];
           atat+=t[i];
         }
        printf("\n\n Job \t Burst Time \t Wait Time \t Turn 
Around Time   Priority \n");
        for(i=0;i<n;i++)
          printf("\n %d \t\t %d  \t\t %d \t\t %d \t\t %d 
\n",p[i],pt[i],w[i],t[i],pp[i]);
        awt/=n;
        atat/=n;
        printf("\n Average Wait Time : %d \n",awt);
        printf("\n Average Turn Around Time : %d \n",atat);
}
```

OUTPUT:



```
pratik@pratik-VirtualBox:~/Desktop/Programs/EXP 4$ gcc pr.c && ./a.out
Enter the number of process : 4

Enter process : time priorities

---- Process no 1----
Priority: 3
Burst time: 1

---- Process no 2----
Priority: 4
Burst time: 2

---- Process no 3----
Priority: 5
Burst time: 3

---- Process no 4----
Priority: 6
Burst time: 4
```



```
Job     Burst Time      Wait Time       Turn Around Time    Priority

4              6               0               6                4

3              5               6               11               3

2              4               11              15               2

1              3               15              18               1

Average Wait Time : 8

Average Turn Around Time : 12
pratik@pratik-VirtualBox:~/Desktop/Programs/EXP 4$ 
```

Shortest Job First

CODE:

```
#include<stdio.h>

int main()
{
	int n,j,temp,temp1,temp2,pr[10],b[10],t[10],w[10],p[10],i;
	float att=0,awt=0;
	for(i=0;i<10;i++)
	{
		b[i]=0;w[i]=0;
	}
	printf("Enter the number of process: ");
	scanf("%d",&n);
	printf("\n---- Burst times ----\n");
	for(i=0;i<n;i++)
	{
		printf("Burst time of Process[%d]: ",i+1);
		scanf("%d",&b[i]);
		p[i]=i;
	}
	for(i=0;i<n;i++)
```

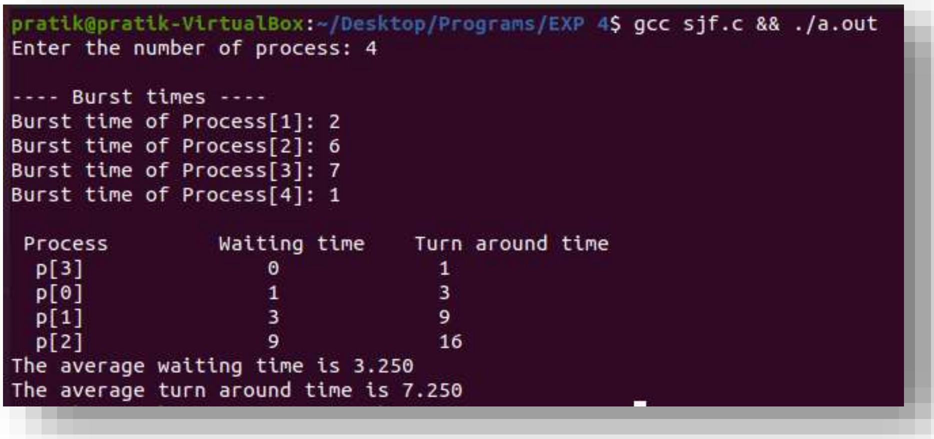
```
    {
        for(j=i;j<n;j++)
        {
            if(b[i]>b[j])
            {
                temp=b[i];
                temp1=p[i];
                b[i]=b[j];
                p[i]=p[j];
                b[j]=temp;
                p[j]=temp1;
            }
        }
    }
    w[0]=0;
    for(i=0;i<n;i++)
    w[i+1]=w[i]+b[i];
    for(i=0;i<n;i++)
    {
        t[i]=w[i]+b[i];
        awt=awt+w[i];
        att=att+t[i];
    }
    awt=awt/n;
    att=att/n;
    printf("\n Process \t Waiting time \t Turn around time
\n");
    for(i=0;i<n;i++)
    printf("  p[%d] \t\t    %d \t\t  %d \n",p[i],w[i],t[i]);
    printf("The average waiting time is %.3f\n",awt);
    printf("The average turn around time is %.3f\n",att);
    return 1;
}
```

Computer Engineering Department &
Information Technology Engineering Department

Academic Year: 2021-2022

Class: S.Y.B.Tech **Sem.:** 4 **Course:** OS

OUTPUT:



```
pratik@pratik-VirtualBox:~/Desktop/Programs/EXP 4$ gcc sjf.c && ./a.out
Enter the number of process: 4

---- Burst times ----
Burst time of Process[1]: 2
Burst time of Process[2]: 6
Burst time of Process[3]: 7
Burst time of Process[4]: 1

 Process         Waiting time    Turn around time
  p[3]               0              1
  p[0]               1              3
  p[1]               3              9
  p[2]               9              16
The average waiting time is 3.250
The average turn around time is 7.250
```

RESULT: I learnt about the different algorithms of CPU scheduling. Algorithms like First Come First Serve, Round Robin, Priority Scheduling, Shortest Job First. Learnt how to take custom user input and pass into functions that calculate the above algorithms.