Exp -1 Name: M. Rohith
Date: 08 -05-2023 Reg no: 3122 215001085

OS LAB TEST -1

Aim:

In the weekend, do 3 tasks, write :

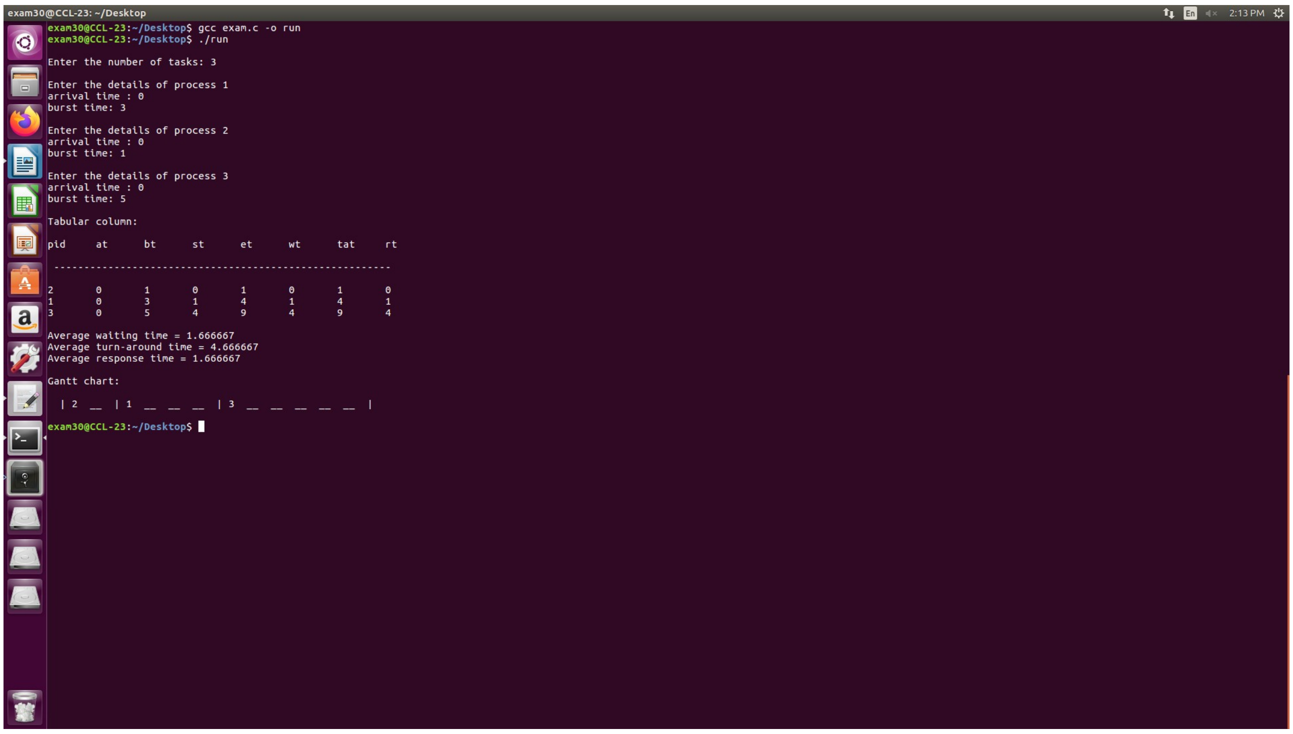
record -3 hours
assignment- 1 hour
project model – 5 hours

All 3 tasks need to be submitted on Monday. Identify the scheduling algorithm and implement the code along with gantt chart. Prepare the i/p and display the o/p.

Program code:

```
//Non-preemptive Shortest Job First

#include <stdio.h>
#include <stdlib.h>

void main()
{
	int pid[10];
	int at[10];
	int bt[10];
	int st[10];
	int et[10];
	int wt[10];
	int tat[10];
	int rt[10];
	int n=3;
	int st_time=0,et_time=0;
	float avg_wt=0,avg_tat=0,avg_rt=0;
	int temp1=0,temp2=0,temp3=0;

	printf("\nEnter the number of tasks: ");
	scanf("%d",&n);

	for(int i=0;i<n;i++)		// getting at and bt
	{
		printf("\nEnter the details of process %d ",i+1);
		pid[i]=i+1;
		printf("\narrival time : ");
		scanf("%d",&at[i]);
		printf("burst time: ");
		scanf("%d",&bt[i]);
	}

	for(int i=0;i<n;i++)		// sorting the pids based on shortest bt
	{
		for(int j=i+1;j<n;j++)
		{
			if(bt[i]>bt[j])
			{
				int temp1=at[i];
				at[i]=at[j];
				at[j]=temp1;

				int temp2=bt[i];
				bt[i]=bt[j];
				bt[j]=temp2;

				int temp3=pid[i];
				pid[i]=pid[j];
				pid[j]=temp3;
			}
		}
	}


	for(int i=0;i<n;i++)		// st is computed
	{
		st[i]=st_time;
		st_time+=bt[i];
	}

	for(int i=0;i<n;i++)		// et is computed
	{
		if(i!=n)
			et[i]=st[i+1];
	}
	et[n-1]=st_time;

	for(int i=0;i<n;i++)		// calculation of wt, tat and rt
	{
		wt[i]=st[i]-at[i];
		tat[i]=wt[i]+bt[i];
		rt[i]=st[i]-at[i];
	}

	printf("\nTabular column: \n");

	printf("\npid\tat\tbt\tst\tet\twt\ttat\trt\n");
	printf("\n ------------------------------------------------------------- \n");

	for(int i=0;i<n;i++)
	{
		printf("\n%d\t%d\t%d\t%d\t%d\t%d\t%d\t
%d",pid[i],at[i],bt[i],st[i],et[i],wt[i],tat[i],rt[i]);
	}
	printf("\n");
	for(int i=0;i<n;i++)
	{
		avg_wt+=wt[i];
		avg_tat+=tat[i];
		avg_rt+=rt[i];
	}


	avg_wt/=n;	//avg wt
	avg_tat/=n;	//avg tat
	avg_rt/=n;	//avg rt

	printf("\nAverage waiting time = %f",avg_wt);
	printf("\nAverage turn-around time = %f",avg_tat);
	printf("\nAverage response time = %f",avg_rt);
	printf("\n");

	printf("\nGantt chart:\n\n ");

	for(int i=0;i<n;i++)
	{
		printf(" | %d ",pid[i]);
		for(int j=st[i];j<et[i];j++)
		{
			printf(" __ ");
		}
	}
	printf(" | \n\n");
}
```

Output:



```
exam30@CCL-23:~/Desktop$ gcc exam.c -o run
exam30@CCL-23:~/Desktop$ ./run

Enter the number of tasks: 3

Enter the details of process 1
arrival time : 0
burst time: 3

Enter the details of process 2
arrival time : 0
burst time: 1

Enter the details of process 3
arrival time : 0
burst time: 5

Tabular column:

pid     at      bt      st      et      wt      tat     rt

 --------------------------------------------------------

2       0       1       0       1       0       1       0
1       0       3       1       4       1       4       1
3       0       5       4       9       4       9       4

Average waiting time = 1.666667
Average turn-around time = 4.666667
Average response time = 1.666667

Gantt chart:

  | 2  __  | 1  __  __  __  | 3  __  __  __  __  __  |

exam30@CCL-23:~/Desktop$
```

Reason:

- Non preemptive SJF is used in the given problem.
- Since before starting any assignment, students used to start their tasks by starting with the task that consumes less amount of time compared to other tasks.
- Non preemptive is used because if a task or assignment is started, we must complete it in order to enhance the accuracy so as to avoid errors by messing up with different assignments.
- Hence non- preemptive SJF best suits for the given problem.

Learning Outcome:

Thus the program code along with the output has been verified.