Write a C program to simulate the following non-preemptive CPU scheduling algorithms to find turnaround time and waiting time. a) FCFS b) SJF c) Round Robin (pre-emptive) d) Priority

c) round robin(pre-emptive)

```
#include<stdio.h>
main()
{
int i,j,n,bu[10],wa[10],tat[10],t,ct[10],max;
float awt=0,att=0,temp=0;
printf("Enter the no of processes -- ");
scanf("%d",&n);
for(i=0;i<n;i++)
{
	printf("\nEnter Burst Time for process %d -- ", i+1);
	scanf("%d",&bu[i]);
	ct[i]=bu[i];
}
printf("\nEnter the size of time slice -- ");
scanf("%d",&t);
max=bu[0];
for(i=1;i<n;i++)
	if(max<bu[i])
		max=bu[i];
	for(j=0;j<(max/t)+1;j++)
		for(i=0;i<n;i++)
			if(bu[i]!=0)
				if(bu[i]<=t)
				{
					tat[i]=temp+bu[i];
					temp=temp+bu[i];
					bu[i]=0;
				}
				else
				{
					bu[i]=bu[i]-t;
					temp=temp+t;
				}
for(i=0;i<n;i++)
{
	wa[i]=tat[i]-ct[i];
	att+=tat[i];
	awt+=wa[i];
	}
printf("\nThe Average Turnaround time is -- %f",att/n);
printf("\nThe Average Waiting time is -- %f ",awt/n);
printf("\n\tPROCESS\t BURST TIME \t WAITING TIME\tTURNAROUND TIME\n");
for(i=0;i<n;i++)
printf("\t%d \t %d \t\t %d \t\t %d \n",i+1,ct[i],wa[i],tat[i]);
getch();}
```

```
Enter the no of processes -- 5

Enter Burst Time for process 1 -- 5

Enter Burst Time for process 2 -- 4

Enter Burst Time for process 3 -- 8

Enter Burst Time for process 4 -- 10

Enter Burst Time for process 5 -- 3

Enter the size of time slice -- 3

The Average Turnaround time is -- 21.200001
The Average Waiting time is -- 15.200000
        PROCESS  BURST TIME      WAITING TIME   TURNAROUND TIME
        1        5               12             17
        2        4               14             18
        3        8               18             26
        4        10               20             30
        5        3               12             15
```
program 2) Write a C program to simulate the MVT and MFT memory management techniques.

MFT

```
#include<stdio.h>
#include<conio.h>
main()
{
int ms, bs, nob, ef,n,
mp[10],tif=0; int i,p=0;
printf("Enter the total memory available (in Bytes) -- ");
scanf("%d",&ms);
printf("Enter the block size (in Bytes) -- ");
scanf("%d", &bs);
nob=ms/bs;
ef=ms - nob*bs;
printf("\nEnter the number of processes -- ");
scanf("%d",&n);
for(i=0;i<n;i++)
{
printf("Enter memory required for process %d (in Bytes)-- ",i+1);
scanf("%d",&mp[i]);
}
printf("\nNo. of Blocks available in memory--%d",nob);
printf("\n\nPROCESS\tMEMORYREQUIRED\tALLOCATED\tINTERNAL FRAGMENTATION");
for(i=0;i<n && p<nob;i++)
{
        printf("\n %d\t\t%d",i+1,mp[i]);
        if(mp[i] > bs)
                printf("\t\tNO\t\t---");
        else
        {
                printf("\t\tYES\t%d",bs-mp[i]);
                tif = tif + bs-mp[i];
                p++;
        }
}
if(i<n)
printf("\nMemory is Full, Remaining Processes cannot be accomodated");
printf("\n\nTotal Internal Fragmentation is %d",tif);
printf("\nTotal External Fragmentation is %d",ef);
getch();
}
```

```
Enter the total memory available (in Bytes) -- 1000
Enter the block size (in Bytes) -- 200

Enter the number of processes -- 5
Enter memory required for process 1 (in Bytes)-- 150
Enter memory required for process 2 (in Bytes)-- 22
Enter memory required for process 3 (in Bytes)-- 200
Enter memory required for process 4 (in Bytes)-- 150
Enter memory required for process 5 (in Bytes)-- 150

No. of Blocks available in memory--5

PROCESS MEMORYREQUIRED  ALLOCATED      INTERNAL FRAGMENTATION
 1          150          YES    50
 2          22          YES    178
 3          200          YES    0
 4          150          YES    50
 5          150          YES    50

Total Internal Fragmentation is 328
Total External Fragmentation is 0
```

MVT:

```
#include<stdio.h>
#include<conio.h>
main()
{
int ms,mp[10],i,
temp,n=0; char ch = 'y';
printf("\nEnter the total memory available (in Bytes)-- ");
scanf("%d",&ms);
temp=ms;
for(i=0;ch=='y';i++,n++)
{
printf("\nEnter memory required for process %d (in Bytes) -- ",i+1);
scanf("%d",&mp[i]);
if(mp[i]<=temp)
{
printf("\nMemory is allocated for Process %d ",i+1);
temp = temp - mp[i];
}
else
{
printf("\nMemory is Full"); break;
}
printf("\nDo you want to continue(y/n) -- ");
scanf(" %c", &ch);
}
printf("\n\nTotal Memory Available -- %d", ms);
printf("\n\n\tPROCESS\t\t MEMORY ALLOCATED ");
for(i=0;i<n;i++)
printf("\n \t%d\t\t%d",i+1,mp[i]);
printf("\n\nTotal Memory Allocated is %d",ms-temp);
printf("\nTotal External Fragmentation is %d",temp);
getch();
}
```

```
Enter memory required for process 1 (in Bytes) -- 300

Memory is allocated for Process 1
Do you want to continue(y/n) -- y

Enter memory required for process 2 (in Bytes) -- 400

Memory is allocated for Process 2
Do you want to continue(y/n) -- y

Enter memory required for process 3 (in Bytes) -- 100

Memory is allocated for Process 3
Do you want to continue(y/n) -- y

Enter memory required for process 4 (in Bytes) -- 300

Memory is Full

Total Memory Available -- 1000

    PROCESS     MEMORY ALLOCATED
    1        300
    2        400
    3        100

Total Memory Allocated is 800
Total External Fragmentation is 200
```

PROGRAM 5: Write a C program to simulate producer-consumer problem using semaphores.

```
#include<stdio.h>
void main()
{
int buffer[10], bufsize, in, out, produce, consume,
choice=0; in = 0;
out = 0;
bufsize = 5;
while(choice !=3)
{
printf("\\n1. Produce \t 2. Consume \t3. Exit");
printf("\nEnter your choice: ");
scanf("%d",&choice);
switch(choice) {
case 1: if((in+1)%bufsize==out)
                        printf("\nBuffer is Full");
                else
                {
                        printf("\nEnter the value: ");
                        scanf("%d", &produce);
                        buffer[in] = produce;
                        in = (in+1)%bufsize;
                }
                        break;
                case 2:
                        if(in == out)
                        printf("\nBuffer is Empty");
                        else
                        {
                        consume = buffer[out];
                        printf("\nThe consumed value is %d", consume);
                        out = (out+1)%bufsize;
                        }
                        break;
                }
        }
        getch();
}
```

```
\n1. Produce     2. Consume     3. Exit
Enter your choice: 1

Enter the value: 200
\n1. Produce     2. Consume     3. Exit
Enter your choice: 1

Enter the value: 300
```

```
\n1. Produce     2. Consume     3. Exit
Enter your choice: 2

The consumed value is 200\n1. Produce    2. Consume     3. Exit
Enter your choice:
```