VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.



LAB REPORT on

Operating Systems

(22CS4PCOPS)

Submitted by:

Vismitha Raj S Doshi (1WA23CS047)

in partial fulfillment for the award of the degree of BACHELOR OF ENGINEERING

COMPUTER SCIENCE AND ENGINEERING



B.M.S. COLLEGE OF ENGINEERING

(Autonomous Institution under VTU)

BENGALURU-560019

Feb 2025 - June 2025

B. M. S. College of Engineering, Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum) **Department of Computer Science and Engineering**



CERTIFICATE

This is to certify that the Lab work entitled "Operating Systems" carried out by Vismitha Raj S Doshi(1WA23CS047), who is a bonafide student of B. M. S. College of Engineering. It is in partial fulfillment for the award of Bachelor of Engineering in Computer Science and Engineering of the Visvesvaraya Technological University, Belgaum during the year 2024-25. The Lab report has been approved as it satisfies the academic requirements in respect of Operating Systems - (22CS4PCOPS) work prescribed for the said degree.

Dr Seema PatilAssociate Professor
Department of CSE
BMSCE, Bengaluru

Dr. Kavitha SoodaProfessor and Head
Department of CSE
BMSCE, Bengaluru

Table Of Contents

Lab program no.	Program	Page no.
1.	FCFS, SJF	6-10
2.	Priority	11-15
3.	Multilevel queue	16-19
4.	Rate Monotonic and earliest deadline	19-25
5.	Producer consumer	29-31
6.	Dining philosopher	32-35
7.	Banker's Algorithm	35-39
8.	Dead detection	39-42
9.	First fit, Best fit, Worst fit	43-48
10.	LRU,FIFO,Optimal	48-55

Index

S.No.	Date	Title	Page No.	Teacher's San
1	6/3/25	FCFS & STC	10	V
2	alslar	Priority (and & non peg	(m)	-
	170	enphra)	0	2032
3	3/4/25	Multhered Queue	1	ON
	3/4/25	Rate Monette		No.
	3/4/25	Earliest diadline		-
9	16/4/25	Produces Consumes		RAN
À	16/4/25	Divising Puttosopher.	TELE	-
6)	17/4/25	banker's Algorithm	14	De L
	ralular	Devollock		24
6	15/5/25	Howary allocation	9.	7,
4	15/21	Page replacement	77)
		FIFO.	-/	
		Leu		1 Sut
		opternal.		150
			A 101	1
		7		
			17 304	100
		The same of the sa	-53	1
	4			
	·	9 7		
	3. 4			
-	1111	***		1.0
		4	1	
		The second second		-61
			185	
		1 1 1 1 1 1		
			kar's/Victory	

Course Outcomes

CO1: Apply the different concepts and functionalities of Operating System.

CO2: Analyse various Operating system strategies and techniques.

CO3: Demonstrate the different functionalities of Operating System.

CO4: Conduct practical experiments to implement the functionalities of Operating system.

GITHUB LINK:

https://github.com/vismitharaj/OS-lab

Experiments

- 1. Write a C program to simulate the following non-pre-emptive CPU scheduling algorithm to find turnaround time and waiting time.
- (a) FCFS
- (b) SJF

```
#include<stdio.h>
  int n, i, j, pos, temp, choice, Burst_time[20], Waiting_time[20],
  Turn_around_time[20], process[20], total=0;
  float avg Turn around time=0, avg Waiting time=0;
int FCFS()
{
  Waiting_time[0]=0;
  for(i=1;i<n;i++)
     Waiting time[i]=0;
    for(j=0;j< i;j++)
       Waiting_time[i]+=Burst_time[i];
  }
  printf("\nProcess\t\tBurst Time\t\tWaiting Time\t\tTurnaround Time");
  for(i=0;i< n;i++)
     Turn around time[i]=Burst time[i]+Waiting time[i];
     avg_Waiting_time+=Waiting_time[i];
     avg Turn around time+=Turn around time[i];
printf("\nP[%d]\t\t%d\t\t\t%d\t\t\t\d",i+1,Burst_time[i],Waiting_time[i],Turn_around_t
ime[i]);
  }
  avg Waiting time =(float)(avg Waiting time)/(float)i;
  avg_Turn_around_time=(float)(avg_Turn_around_time)/(float)i;
  printf("\nAverage Waiting Time:%.2f",avg_Waiting_time);
  printf("\nAverage Turnaround Time:%.2f\n",avg_Turn_around_time);
  return 0;
}
```

```
int SJF()
  //sorting
  for(i=0;i< n;i++)
     pos=i;
     for(j=i+1;j< n;j++)
       if(Burst_time[j]<Burst_time[pos])
          pos=j;
     }
     temp=Burst_time[i];
     Burst_time[i]=Burst_time[pos];
     Burst_time[pos]=temp;
     temp=process[i];
     process[i]=process[pos];
     process[pos]=temp;
  }
     Waiting_time[0]=0;
  for(i=1;i<n;i++)
     Waiting_time[i]=0;
     for(j=0;j< i;j++)
       Waiting_time[i]+=Burst_time[j];
     total+=Waiting_time[i];
  }
  avg_Waiting_time=(float)total/n;
  total=0;
  printf("\nProcess\t\tBurst Time\t\tWaiting Time\t\tTurnaround Time");
  for(i=0;i< n;i++)
     Turn_around_time[i]=Burst_time[i]+Waiting_time[i];
     total+=Turn_around_time[i];
printf("\nP[%d]\t\t%d\t\t\t%d\t\t\t%d",process[i],Burst_time[i],Waiting_time[i],Turn_ar
ound_time[i]);
```

```
}
  avg_Turn_around_time=(float)total/n;
  printf("\n\nAverage Waiting Time=%f",avg_Waiting_time);
  printf("\nAverage Turnaround Time=%f\n",avg_Turn_around_time);
}
int main()
  printf("Enter the total number of processes:");
  scanf("%d",&n);
  printf("\nEnter Burst Time:\n");
  for(i=0;i< n;i++)
     printf("P[%d]:",i+1);
     scanf("%d",&Burst_time[i]);
     process[i]=i+1;
  }
  while(1)
  { printf("\n----\n");
     printf("1. FCFS Scheduling\n2. SJF Scheduling\n");
     printf("\nEnter your choice:");
     scanf("%d", &choice);
     switch(choice)
       case 1: FCFS();
       break;
       case 2: SJF();
       break;
       default: printf("Invalid Input!!!");
     }
  }
  return 0;
}
```

a.

```
ArrivalTime.c -0 FCFS ArrivalTime }; if ($?) { .\FCFS ArrivalTime }
 Enter the number of processes: 4
 Enter the process ids:
 1234
 Enter arrival time and burst time for process 1: 0 8
 Enter arrival time and burst time for process 2: 1 4
 Enter arrival time and burst time for process 3: 2 9
 Enter arrival time and burst time for process 4: 3 5
                                                      Turnaround Time
 Process Arrival Time Burst Time
                                       Waiting Time
                        8
                                       0
         1
                        4
                                                      11
         2
                        9
                                                       19
                                       10
 4
                                       18
                                                       23
 Average Waiting Time: 8.75
 Average Turnaround Time: 15.25
O PS C:\Users\Nisarga Gondi\OneDrive\Desktop\Nisarga\IV SEM\OS 4th sem\os lab>
```

b.

```
P.c -0 SJF_NP \}; if (\S?) \{ .\SJF_NP \} Enter the number of processes:
Enter the burst time of process 1:
Enter the burst time of process 2:
4
Enter the burst time of process 3:
9
Enter the burst time of process 4:
5
BurstTime
                WaitingTime
                                  TurnAroundtime
4.00
                0.00
                                  4.00
5.00
                 4.00
                                  9.00
                                  17.00
8.00
                9.00
                17.00
9.00
                                  26.00
Average waiting time:7.500000
Average turn around time:14.000000
```

```
Lab Program 1
    Write a c program to 1200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 200 | 20
                                                                             non . per - emptone
                                 11) SIF
    ?) FCFS
    Africade < etclio, N
    # Puchedo c etalish. N>
   A defens max 10
    Stract persons &
    int id, AT, BT, CT, TAT, WT, RT, semaining-BT;
     Put completed;
         Threeting > 70 (779 24 botoly was E/2)
   Void Port-by-AT (Proud process pc), int n) [
     Par (int 1=0; icn p a-1; i++) [1] - Table
           for (int j=1+1; kn; j++) ?
              2 (TA. [139 < TA. [1797]
                     stout prouse temp = p(i);
                      : [i79 = [i79
              P 557 = tourp;
     Void calculate_FCFS Count process pro, int no
       Sort by AT (p, h);
       Put wormethine = 0; 10 1 1 to the total of
       Pad (int i=0; izn; i++)
         ef Casantilus < p(i) AT)
                           TA. Eigg = surstance
 void calculate - STF - framptine
     ( prent from pe), int n) c
   ind completed = 0, corrent since = 0 (100) /200
    for (intico; icn; 1+0)
      Ridremaining-Bt = Prid BT
   while (completed < n) = 10000;
       2(++i; n) i; 0 = i tri) to3
         of (ipri). completed 21 ptiJ.AT <=
         3(Tanins Ta-grin somer. [179 ble enithiscon
         number = PSi]. romaining-BT;
     2 (1-== trotlants) }9
              (++ smitting ++)
        if (p [sucrent], remaining-BT == 
P[sucrent] 2787
            PIShpetent 7.27 = warrentime -
         pt. shortest J. AT;
          ¿ = 18-priniane. [ Istarbana ;
            : ++ emittueres
           If Co Canadost J. evenaining BT == 0)C
            permantent. CT = custemations;
             - TAT. [trateone79 = 7AT. [trateone79
             Princitest]. A.T. = princitest] FAT -
             p[shaltert]. RST.
p[shaltert]. Completed = 1;
completed + 1
33 3 Gold
```

```
PTIJ. RT = cusantiono - PTIT. AT | Paga Na
pri).ct = assouttone + prij. BT; Daloj
 cuseustine = ptil ct
 (TA- [179 - D. [179 = TAT [179
 PETJ.WT = PETJ.TAT - PETJ.BT;
Void calculate- STF_ Non Accomptence ( struct
 Process PIJ, into)C
 Put completed =0; conventione = 0;
 That completed CN) [
  int shortest =- 1, wend = 10000;
   for (int i=0; icn; itt) [
    if (! pri] completed && pri]. AT <= www.
     RE PTJ.BT < NUMBT) [
MINBT = PTJ.BT)
      Phostest = ij
  of (mostert = = -1)c
       assent Time ++:
  I else L
   pernortent]. RT: currentime - pronortest] AT;
   PERMONEUT, RECT = CONSONTTIME & PERMONTENT BT.
   Custometine: promotents. a;
   P[markest]. TAT = P(shortest? of -
                   TA GESTRONS 29
   P [snortest]. WT = P[snortest]. FDT -
                    p [shatox] 87;
    Pishertost completed = 1;
    completed ++;
 Output
 Enter no. of procurses = 4

Buter Armival teno (AT): 00200
 Enter Bust Thus (BT): 2 4 86 8
  FCTE
Process OF AT BT TAT
          6 1 6 2 9
            20. 20 12 12
                 6
                00
```

2.Priority

```
#include <stdio.h>
#define MAX 10
typedef struct {
  int pid, at, bt, pt, remaining_bt, ct, tat, wt, rt, is_completed, st;
} Process;
void nonPreemptivePriority(Process p[], int n) {
  int time = 0, completed = 0;
  while (completed < n) {
     int highest_priority = 9999, selected = -1;
     for (int i = 0; i < n; i++) {
        if (p[i].at <= time && !p[i].is_completed && p[i].pt < highest_priority) {
          highest_priority = p[i].pt;
          selected = i;
        }
     if (selected == -1) {
        time++;
        continue;
     if (p[selected].rt == -1) {
        p[selected].st = time; // Start time
        p[selected].rt = time - p[selected].at;
     time += p[selected].bt;
     p[selected].ct = time;
     p[selected].tat = p[selected].ct - p[selected].at;
     p[selected].wt = p[selected].tat - p[selected].bt;
     p[selected].is_completed = 1;
     completed++;
  }
}
void preemptivePriority(Process p[], int n) {
  int time = 0, completed = 0;
  while (completed < n) {
     int highest_priority = 9999, selected = -1;
     for (int i = 0; i < n; i++) {
        if (p[i].at <= time && p[i].remaining_bt > 0 && p[i].pt < highest_priority) {
          highest_priority = p[i].pt;
          selected = i;
        }
```

```
if (selected == -1) {
        time++;
        continue;
     if (p[selected].rt == -1) {
        p[selected].st = time; // Start time
        p[selected].rt = time - p[selected].at;
     }
     p[selected].remaining_bt--;
     time++;
     if (p[selected].remaining_bt == 0) {
        p[selected].ct = time;
        p[selected].tat = p[selected].ct - p[selected].at;
        p[selected].wt = p[selected].tat - p[selected].bt;
        completed++;
     }
  }
}
void displayProcesses(Process p[], int n) {
  float avg_tat = 0, avg_wt = 0, avg_rt = 0;
  printf("\nPID\tAT\tBT\tPriority\tCT\tTAT\tWT\tRT\n");
  for (int i = 0; i < n; i++) {
     printf("%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\n",
          p[i].pid, p[i].at, p[i].bt, p[i].pt, p[i].ct, p[i].tat, p[i].wt, p[i].rt);
     avg tat += p[i].tat;
     avg_wt += p[i].wt;
     avg_rt += p[i].rt;
  }
  printf("\nAverage TAT: %.2f", avg_tat / n);
  printf("\nAverage WT: %.2f", avg_wt / n);
  printf("\nAverage RT: %.2f\n", avg_rt / n);
}
int main() {
  Process p[MAX];
  int n, choice;
  printf("Enter the number of processes: ");
  scanf("%d", &n);
  for (int i = 0; i < n; i++) {
     p[i].pid = i + 1;
```

```
printf("\nEnter Arrival Time, Burst Time, and Priority for Process %d:\n", p[i].pid);
  printf("Arrival Time: ");
  scanf("%d", &p[i].at);
  printf("Burst Time: ");
  scanf("%d", &p[i].bt);
  printf("Priority: ");
  scanf("%d", &p[i].pt);
  p[i].remaining_bt = p[i].bt;
  p[i].is\_completed = 0;
  p[i].rt = -1;
}
while (1) {
  printf("\nPriority Scheduling Menu:\n");
  printf("1. Non-Preemptive Priority Scheduling\n");
  printf("2. Preemptive Priority Scheduling\n");
  printf("3. Exit\n");
  printf("Enter your choice: ");
  scanf("%d", &choice);
  switch (choice) {
     case 1:
        nonPreemptivePriority(p, n);
        printf("Non-Preemptive Scheduling Completed!\n");
        displayProcesses(p, n);
        break;
     case 2:
        preemptivePriority(p, n);
        printf("Preemptive Scheduling Completed!\n");
        displayProcesses(p, n);
        break;
     case 3:
        printf("Exiting...\n");
        return 0;
     default:
        printf("Invalid choice! Try again.\n");
  }
}
return 0;
```

}

```
Enter the number of processes: 7
Enter Arrival Time, Burst Time, and Priority for Process 1:
Arrival Time: 0
Burst Time: 8
Priority : 3
Enter Arrival Time, Burst Time, and Priority for Process 2:
Arrival Time: 1
Burst Time: 2
Priority : 4
Enter Arrival Time, Burst Time, and Priority for Process 3:
Arrival Time: 3
Burst Time: 4
Priority : 4
Enter Arrival Time, Burst Time, and Priority for Process 4:
Arrival Time: 4
Burst Time: 1
Priority : 5
Enter Arrival Time, Burst Time, and Priority for Process 5:
Arrival Time: 5
Burst Time: 6
Priority : 2
Enter Arrival Time, Burst Time, and Priority for Process 6:
Arrival Time: 6
Burst Time: 5
Enter Arrival Time, Burst Time, and Priority for Process 6:
Arrival Time: 6
Burst Time: 5
Priority : 6
Enter Arrival Time, Burst Time, and Priority for Process 7:
Arrival Time: 10
Burst Time: 1
Priority : 1
Priority Scheduling Menu:

    Non-Preemptive Priority Scheduling

Preemptive Priority Scheduling
3. Exit
Enter your choice: 1
Non-Preemptive Scheduling Completed!
PID
1
2
       AT
               ВТ
                       Priority
                                       CT
                                               TAT
                                                      WT
                                                               RT
       ø
               8
                                               8
                                                       Ø
                                                              ø
                       3
                                       8
                                       17
       1
               2
                                               16
                                                       14
                                                               14
3
                                               18
       3
                                       21
                                                       14
                                                               14
               1
                       5
                                       22
                                               18
                                                       17
                                                               17
               6
                                       14
                                               9
6
                       6
                                       27
                                               21
                                                       16
                                                               16
        6
        10
                                       15
                                               5
                                                       4
                                                               4
```

```
Labo Program
Wolte a c purgram to 9thrubate Page No. / 200
con scheduling for processors (100) /200 to fend turn around time and walking time
 using potosty (Preemptive and non pot empton)
# Prichale < PHOO. N.
# degene max 10
typedef struct &
       Put pid, at, bt, pt, semalaryg-bt, ct, tat, wt,
         8t, 9s_completed, st;
Void nonfreempth ve Pressity (Process P(), End n) [
     Put teme = 0, completed =0;
     while (completed < n) [
            Int heghest-priority = 9999, selected = . - 1;
               for( fut ?=0; ich; i+t) [
                 of (pri) at < = time && ! pri] is completed
                   & & pOD. pt < heggest periority) [
                     highest-priority = pli]. pt;
                         selected = ?;
            of (seloched == -DI
                          tene ++:
                          continue;
              Pt (predocted) st == -1) [
                       p[edected].st = tenne;
                       predected Tot Home - predected T. at,
              temet = pssolected ]. bt;
     ong-tati - pro row,
     ang_wt= ptil. wt;
ang_rt+= ptil. ct;
prent f("Average TAT: 12f", ang-too (n);
 perent ( " Amage wit : 1.024", ang -w+ (n)
 perent (" In Amongo RT 1.2f in", ang of In)
 Put mach () &
     Process P[Max];
     Put n, choice:
     printf ( Enter the no. of purases:"
     Scant (" i'd", 20);
     forci=orien;i++)2.
      Prid= 1+1; P
      postfl" Enter for asseral time, bush time as
     Presently for process "1d; ", pos. pid);
     Rent (" Anival Hous ").
                                                                     1 F.P 7W 3page
     prentf(" Burst teme ");
    Scauf ("Id", & PIJ (t));
percent (" Potority");
scauf ("Id; & PIJ. et);
                                                                       himbour pin
   phil remarking to prish by it
     pti] . sut = -1;01
      while(1) E
              previte ("Britority Schadding Manus")

previte ("Britority Schadding Manus")

previte ("Britority")

previte ("Britority Schadding Manus")

previte ("Brito
               swetche ceroico ?
                           non Procupero Privarity (pm)
              Gold pront ("Non prompting); sheller / Victory
```

```
p[selected]. ct = teme;
P[soloted]. tat: P[soloted]. t-
             predocted J. at;
 presented , wt = predected that - predected to bt.
 P[selovered] is_completed = 1;
 completed;
void preemptevelionity (Process PC), int n) (
   Int three o, completed = 0;
   while (completed < n) (
     Put highert Priority = 9999, selected = -1;
for (but i=0; icn; i++) [
      If (prilat <= 19me ed pris ramalning bt >0 28
       PCIT. pt < heghest-paiority) (
          highest - Periority - ptis. pt;
       selected = 1;
    if (selected = = - 1) ?
         thrue ++
         continue:
    9 (0== td-grincanor. Theseses) 79
        p [solected]. It = Here
        producted tat = producted to - producted at;
        p [ redocted] wt = p (redocted) tat - predected), bt
        completed + +;
void display frounds (Procurs of I), Put n) &
  plant. aug_tat=0; aug_wt=0; aug_t-0;
   pullet ("In 810 to AT) + BT It Briority to CT ItTAK &
        tWT ITRT In");
  for (i=0; icn; i++) &
      displayfrowss(p, N)
  precuptive (p,n)
display froum (p,n)
 default: publific "Invalid choke");
 Enter the no of persusses: 7
 output:
                                              RPRT
                   Priority
     AT
                    300
                                          16
                                                14
       0
```

4 4 17 17 22 14 27 Average TAT: 13.57 Average WT: 9.71 Broring scheduling Entre your choice PID AT BT Bion BT Priority TAT WT CT 0 14 14 DI 17 02 12 7 2) 10 Average 7AT: 13.71 Average WT: 9.86 Amore P3 P3 P1 (P2 P3) P4

3.Write a C program to simulate multi-level queue scheduling algorithm considering the following scenario. All the processes in the system are divided into two categories – system processes and user processes. System processes are to be given higher priority than user processes. Use FCFS scheduling for the processes in each queue.

```
#include <stdio.h>
#define MAX PROCESSES 10
#define TIME_QUANTUM 2
typedef struct {
  int burst_time, arrival_time, queue_type, waiting_time, turnaround_time,
response_time, remaining_time;
} Process:
void round_robin(Process processes[], int n, int time_quantum, int *time) {
  int done, i:
  do {
     done = 1;
     for (i = 0; i < n; i++) {
       if (processes[i].remaining_time > 0) {
          done = 0:
          if (processes[i].remaining_time > time_quantum) {
             *time += time_quantum;
            processes[i].remaining_time -= time_quantum;
          } else {
             *time += processes[i].remaining_time;
             processes[i].waiting_time = *time - processes[i].arrival_time -
processes[i].burst_time;
             processes[i].turnaround time = *time - processes[i].arrival time;
             processes[i].response_time = processes[i].waiting_time;
            processes[i].remaining_time = 0;
       }
  } while (!done);
void fcfs(Process processes[], int n, int *time) {
  for (int i = 0; i < n; i++) {
     if (*time < processes[i].arrival_time) {
       *time = processes[i].arrival_time;
     }
     processes[i].waiting_time = *time - processes[i].arrival_time;
     processes[i].turnaround_time = processes[i].waiting_time +
processes[i].burst time;
     processes[i].response_time = processes[i].waiting_time;
     *time += processes[i].burst_time;
  }
}
```

```
int main() {
  Process processes[MAX_PROCESSES], system_queue[MAX_PROCESSES],
user queue[MAX PROCESSES];
  int n, sys_count = 0, user_count = 0, time = 0;
  float avg waiting = 0, avg turnaround = 0, avg response = 0, throughput;
  printf("Enter number of processes: ");
  scanf("%d", &n);
  for (int i = 0; i < n; i++) {
     printf("Enter Burst Time, Arrival Time and Queue of P%d: ", i + 1);
    scanf("%d %d %d", &processes[i].burst_time, &processes[i].arrival_time,
&processes[i].queue_type);
     processes[i].remaining time = processes[i].burst time;
     if (processes[i].queue_type == 1) {
       system_queue[sys_count++] = processes[i];
    } else {
       user_queue[user_count++] = processes[i];
    }
  for (int i = 0; i < user count - 1; i++) {
    for (int i = 0; i < user\_count - i - 1; i + +) {
       if (user queue[j].arrival time > user queue[j + 1].arrival time) {
         Process temp = user_queue[j];
          user queue[i] = user queue[i + 1];
         user_queue[j + 1] = temp;
       }
    }
  printf("\nQueue 1 is System Process\nQueue 2 is User Process\n");
  round_robin(system_queue, sys_count, TIME_QUANTUM, &time);
  fcfs(user_queue, user_count, &time);
  printf("\nProcess Waiting Time Turn Around Time Response Time\n");
  for (int i = 0; i < sys count; i++) {
    avg_waiting += system_queue[i].waiting_time;
     avg_turnaround += system_queue[i].turnaround_time;
     avg_response += system_queue[i].response_time;
     printf("%d
                               %d
                                           %d\n", i + 1,
                    %d
system_queue[i].waiting_time, system_queue[i].turnaround_time,
system queue[i].response time);
  }
  for (int i = 0; i < user\_count; i++) {
     avg waiting += user gueue[i].waiting time;
```

```
avg_turnaround += user_queue[i].turnaround_time;
    avg_response += user_queue[i].response_time;
    printf("%d
                    %d
                               %d
                                           %d\n", i + 1 + sys count,
user_queue[i].waiting_time, user_queue[i].turnaround_time,
user queue[i].response time);
  }
  avg_waiting /= n;
  avg turnaround /= n;
  avg_response /= n;
  throughput = (float)n / time;
  printf("\nAverage Waiting Time: %.2f", avg_waiting);
  printf("\nAverage Turn Around Time: %.2f", avg_turnaround);
  printf("\nAverage Response Time: %.2f", avg_response);
  printf("\nThroughput: %.2f", throughput);
  printf("\nProcess returned %d (0x%d) execution time: %.3f s\n", time, time,
(float)time);
  return 0;
```

}

```
"C:\Users\Admin\OS- WACS47\mq.exe"
Enter number of processes: 4
Enter Burst Time, Arrival Time and Queue of P1: 2 0 1
Enter Burst Time, Arrival Time and Queue of P2:
Enter Burst Time, Arrival Time and Queue of P3: 5 0 1
Enter Burst Time, Arrival Time and Queue of P4: 3 0 2
Queue 1 is System Process
Queue 2 is User Process
Process Waiting Time Turn Around Time Response Time
                                        0
          0
                       2
          2
                       8
          8
                       11
Average Waiting Time: 4.25
Average Turn Around Time: 7.00
Average Response Time: 4.25
Throughput: 0.36
Process returned 11 (0x11) execution time: 11.000 s
Process returned 0 (0x0)
                           execution time : 21.577 s
Press any key to continue.
```

```
algorithm consider the following even and people than consider the following eare divided and the processes on the system processes and the processes of the processes of the processes are to be two categories - system processes are to be that two categories - system processes are to be that two categories processes were processes when the processes were processes to the processes and the processes are processes and the processes are the processes are the processes and the processes are the pro
                                                                                                                                           I while ! done );
                                                                                                                                     void fefs (Process processes () But n, int
                                                                                                                                          for (Put 1=0; icn; == 1+1)e
                                                                                                                                             onthe levines (2) as es oug - Ano) [
                                                                                                                                                   * Home = fewers est arrival_temo;
                                                                                                                                             processes [] walking three = theme - processes at
                                                                                                                                           processes (i) turnoround tems - processes (i) asserting tout
                                                                                                                                                       procuses [i] burst terre;
                                                                                                                                            procured 3 supone thre = perous est 7 walting things
  # Puchade cetalio. h>
 # defens now founder 10
 # defere Terre-Quantum 2
   type dof growt &
      but must there, arrival terms, queue type,
      waiting the turnal and the personne them, remaining there from
                                                                                                                                        Output:
                                                                                                                                      Enter number of perocessos: 4
                                                                                                                                      Fired Burst TPine, Aubral 1800 and Queux of P!
                                                                                                                                        20 201
                                                                                                                                     Enter Burst thre, Assival three and avere of P2:
  typedof etructif
              round roben (Prount prouses [), Int n, ent
   tems quantum, ent * tems) &
                                                                                                                                     Enser Burst thus, Arrival Home and Quene of P3;
                                                                                                                                      Enter Burt Hous, period there and Queue of Pa;
   Put done, 9;
   2 alo
          done : 1:
                                                                                                                                           302
                                                                                                                                        Queue 1 15 Syrtom Prous
Queue 2 15 Uses Prous
           Par (P=0; icn; iti) [
              of (puparti). something- Here > 0)?
                                                                                                                                         Process waiting time turnaround rince Perpanse this
                   done =0;
               Pf (prounos (i) somaling the & temo-quantum
                   Atems += 1800 - quandum;
                     process estil. aumorbing the = + there quantum
                                                                                                                                                                                                      11
                                                                                                                                        Average Waiting Here: 4.25
Average Tarn Around Here: 7.00
                  ( engl princement ( transcrip = + ung +
                  prounesti) wattry three = * that - procured of

- procured to burt some;
                                                                                                                                       Average Response Home : 4.25
                    processed: easyonx tems = processes () wt;
                                                                                                                                          Moughput 0.36
      , , ? personsossis remaining tems =0;
```

4. Write a C program to simulate Real-Time CPU Scheduling

algorithms: a) Rate- Monotonic b) Earliest-deadline First

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <stdbool.h>

#define MAX_PROCESS 10

int num_of_process;

int execution_time[MAX_PROCESS], period[MAX_PROCESS],
 remain_time[MAX_PROCESS], deadline[MAX_PROCESS],
 remain_deadline[MAX_PROCESS];

```
void get_process_info() {
  printf("Enter total number of processes (maximum %d): ", MAX_PROCESS);
  scanf("%d", &num_of_process);
  if (num_of_process < 1) {</pre>
     exit(0);
  }
for (int i = 0; i < num_of_process; i++) {
     printf("\nProcess %d:\n", i + 1);
     printf("==> Execution time: ");
     scanf("%d", &execution_time[i]);
     remain_time[i] = execution_time[i];
     printf("==> Period: ");
     scanf("%d", &period[i]);
  }
}
int max(int a, int b, int c) {
  int max;
  if (a >= b \&\& a >= c)
     max = a;
  else if (b >= a \&\& b >= c)
     max = b;
  else if (c \ge a \& c \ge b)
```

```
max = c;
   return max;
}
void print_schedule(int process_list[], int cycles) {
   printf("\nScheduling:\n\n");
   printf("Time: ");
  for (int i = 0; i < cycles; i++) {
     if (i < 10)
        printf("| 0%d ", i);
      else
        printf("| %d ", i);
  }
  printf("|\n");
for (int i = 0; i < num\_of\_process; i++) {
     printf("P[%d]: ", i + 1);
     for (int j = 0; j < cycles; j++) {
        if (process_list[j] == i + 1)
           printf("|####");
       else
           printf("|
                     ");
     }
     printf("|\n");
  }
```

```
}
void rate_monotonic(int time) {
   int process_list[100] = {0}, min = 999, next_process = 0;
  float utilization = 0;
for (int i = 0; i < num_of_process; i++) {
     utilization += (1.0 * execution_time[i]) / period[i];
  }
int n = num_of_process;
  float m = n * (pow(2, 1.0 / n) - 1);
   if (utilization > m) {
     printf("\nGiven problem is not schedulable under the said scheduling
algorithm.\n");
  }
  for (int i = 0; i < time; i++) {
     min = 1000;
     for (int j = 0; j < num\_of\_process; j++) {
        if (remain_time[j] > 0) {
           if (min > period[j]) {
             min = period[j];
             next_process = j;
           }
        }
     }
```

```
if (remain_time[next_process] > 0) {
       process_list[i] = next_process + 1;
       remain_time[next_process] -= 1;
     }
     for (int k = 0; k < num_of_process; k++) {
       if ((i + 1) \% period[k] == 0) {
          remain_time[k] = execution_time[k];
          next_process = k;
       }
     }
  }
  print_schedule(process_list, time);
}
int main() {
  int observation_time;
  get_process_info();
  observation_time = max(period[0], period[1], period[2]);
  rate_monotonic(observation_time);
  return 0;
}Output:
```

```
. Rate Monotonic
. Earliest Deadline first

    Proportional Scheduling

Enter your choice: 1
Enter total number of processes (maximum 10): 3
Process 1:
=> Execution time: 3
=> Period: 20
Process 2:
-> Execution time: 2
-> Period: 5
Process 3:
 > Execution time: 2
=> Period: 10
Scheduling:
rime: | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19
         P[1]: |
P[2]: |####|####|
                       **** | **** |
                                                                        [####]####]
P[3]: |
              14848148481
```

```
Late Nanotoule AND Earliest Deadlers
#Puchadec statio. hs
# defene wax- Processes 10
typedef struct L
  the id;
  Put bt
   but period;
   Put remaining the;
   Put rest-deadletie;
Thrown:
2(n tus, E) company person procures (3, 84 n) & for that i = 0; kn; (1, 1, 1, 1)
   Real (Put j=0 ; j < m + i -1 ; j++) (
broans of 1 = broans (1);

broans of 1 = broans (1);

broans of 1 = broans (1);

1315
     Et (process es[]] period > prio axes [][+1] period) (
Put ged (Put a) Putb) (
  return b=09 a: gcd (b, a7.b);
 Put LCM (Put a, intb) ?
   retworkarblyed (a,b);
 Put calculate Lin (Process processed) , Ent n) [
   Put sesuet = passing ( for period)
   Par (int 1=1; len; 1+0 9
     ( ( Lourse - tencernet, prouses Pi ]. period);
 I return result;
double utilization factor (Process processes (), into)s
  double sum=0:
  for Cent 1=0; icn; 1++) [
     Sum += (double) processori I bt/processorii) pri
   return sum;
3
double must every live n)e
```

```
return n * (pour (a.o.,1.0) (n)-1).
                                        Page No.
Votel rate monotonic- scheduling (
   Process. publishes [], int n)(
  Put Lin. period = calculate_lin (processes, n);
  proute ( "I cm = 1.d mm", com-period);
  putate ("PID Burst Beriod (n"); of "N");
  for (Put i=0; i=n , i+1)e
    pulmpe "/d /d /d/n"), pulmesti] id,
 ? perouses(i) bt, process es(i) period), all
double utilization = utilization_factor (procuse.in)
double threshod = hens-thrushold) (n)
prents ("In1.69 <= 1.69 => 1.8 \n", utilization,
knowshold, (utilitation <= floreshold)?"true": "false");
it (philitation > thoushold) C
  points (" system may not be schedulable 1/1").
  seturn.
Put puline = 0, executed =0; while (Knuline clan period)
  int edected = -1;
  PONCINETIED; icn; 1++) C
   of Chambine 1. procuses [1] period = -0) C
     processessi 3. surrounding to me : promes EiJ. bt.
  Pf (prouses(; I remaining + Euro > 0) (
    selected = i
    break;
 E ( Releated! = -1) [
  printf ("Terre V. d: Poouss V.d is summery In",
   fondine, persons (selected). id);
   perocesses [ selected] securating time --;
   + + berusas
9 else &
```

```
pronte ("Tenne" L'd: CPU is idle ", Hende
Even the no. of precesses: 3
         the three periods:
 PID
                     period
         Burst
```

Earliest Deadline

```
#include <stdio.h>
int gcd(int a, int b) {
  while (b != 0) {
     int temp = b;
     b = a \% b;
     a = temp;
  return a;
}
int lcm(int a, int b) {
  return (a * b) / gcd(a, b);
}
struct Process {
  int id, burst_time, deadline, period;
};
void earliest_deadline_first(struct Process p[], int n, int time_limit) {
  int time = 0;
  printf("Earliest Deadline Scheduling:\n");
  printf("PID\tBurst\tDeadline\tPeriod\n");
  for (int i = 0; i < n; i++) {
     printf("%d\t%d\t\t%d\n", p[i].id, p[i].burst_time, p[i].deadline, p[i].period);
```

```
}
  printf("\nScheduling occurs for %d ms\n", time_limit);
  while (time < time_limit) {
     int earliest = -1;
     for (int i = 0; i < n; i++) {
        if (p[i].burst\_time > 0) {
           if (earliest == -1 || p[i].deadline < p[earliest].deadline) {
             earliest = i;
           }
        }
     }
     if (earliest == -1) break;
     printf("%dms: Task %d is running.\n", time, p[earliest].id);
     p[earliest].burst_time--;
     time++;
  }
}
int main() {
  int n;
  printf("Enter the number of processes: ");
  scanf("%d", &n);
  struct Process processes[n];
  printf("Enter the CPU burst times:\n");
  for (int i = 0; i < n; i++) {
     scanf("%d", &processes[i].burst_time);
     processes[i].id = i + 1;
  }
  printf("Enter the deadlines:\n");
  for (int i = 0; i < n; i++) {
     scanf("%d", &processes[i].deadline);
  }
  printf("Enter the time periods:\n");
  for (int i = 0; i < n; i++) {
     scanf("%d", &processes[i].period);
  }
  int hyperperiod = processes[0].period;
  for (int i = 1; i < n; i++) {
```

```
hyperperiod = lcm(hyperperiod, processes[i].period);
}

printf("\nSystem will execute for hyperperiod (LCM of periods): %d ms\n",
hyperperiod);

earliest_deadline_first(processes, n, hyperperiod);

return 0;
}
```

```
Enter the number of processes: 3
Enter the CPU burst times:
2 3 4
Enter the deadlines:
1 2 3
Enter the time periods:
1 2 3
System will execute for hyperperiod (LCM of periods): 6 ms
Earliest Deadline Scheduling:
PID
        Burst
                Deadline
                                 Period
1
        2
                         1
                                          1
                                          2
2
        3
                         2
3
        4
                         3
                                          3
Scheduling occurs for 6 ms
Oms: Task 1 is running.
1ms: Task 1 is running.
2ms: Task 2 is running.
3ms: Task 2 is running.
4ms: Task 2 is running.
5ms: Task 3 is running.
Process returned 0 (0x0)
                            execution time : 14.084 s
Press any key to continue.
```

```
Earliest doodlens forst scheduling
# Proclude < stalo. h>
Put gcd ( ent a , 9 mts) [
 2(0=10) Simu
  put temp = b;
   b= a1.b;
   ar=temp;
  suturn a;
Ent Lew (But a, Inth) C
  seturn (a+b) (gcd(a1b);
PUE LOW (Put a, Put b) [
   Seturn (a b) /gcd (a16)
struct process l
 gut id, bt, deadline, period
Void consider deadline first (struct Process PTD);
  Portf(" Forbiest deadline Scheduling");
portf(" Forbiest deadline Scheduline to period (n'))
    prents ("Y.d It Y.d \t Y.d \t Y.d \t Y.d \t');
   for ( out i=0; icn; i+1) (-)
     pros.id, pril st. pril dedure, pri ]. period);
 prentf ("In Scheduleng occurs for yed ms/n"),
   time (smit);
 while (time < time ! thruit) [
   1ut earliest =-1;
   parcent i=0; icn; i++) &
     2 (05 76 Eigg) A
         of (carrier == -1 11 pri) deadline <
              . praniert ]. deadline) [
```

```
output:
Enter the mo. of poweres: 3
British the CPU burst thus !
2 4 6
Enter the deadlbut.
                 123
s/or well execute for hyperperiod ( con of
period): 6 ms
                        period
             peadline
      Burst
PID
 .
          occurs for 6ms:
Schedulerg
     Task 1 is runnery
     · Tousk 1
2ms : Forks
    : Taske -
3ms
4ms . Took2 -
      Tooks they by courses they
 personal the mount had to explain strains
```

```
is trailes
23
Ef (earliest = = -1)
       break;
    prentf(" /d my: Tank /d is Funning . In
    P[earliest]. Ed);
    Preordiest ] . St a - -;
    18me ++;
 I conson the
  but u.
   begate ( "I ever for uo d beneaver ")
  Scanf ("1.2", 2 n);
  smut proun. prouncit; (10)
   persetf ("Enter the CPU burst time, in);
    for (?ut i = 0; i < n; i + 1) [
    : Scanf ( " ? d" , & prouns ? ] bt);
     perousses[i]. Pd = 1+1;
   prentf("Enter the dealers (n");
   g (++1; n>1; 0=1 tur) peg
     Stoup("1.0", & processes(i], deadlene);
   prontf("Enter the serve period");
   feat (Put 1:0; ich; 1+4) ?
     Scanf (" 7d. procest? period)
 } Put hyper pariod = paramerli3 period
    for ( Put 8 = 1 ; icn; i++) {
     hyperperiod = Lan (hyperperiod, procures li? pai
   bent ( " som win execute for interborged (ccw
    of periods) : /dme (n) hyperperiod);
   contiest another ferst (proon, n, hyperod)
  "O mader
```

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

```
#include <stdio.h>
#include <stdlib.h>
int mutex = 1, full = 0, empty = 3, x = 0;
void producer();
void consumer();
int wait(int);
int signal(int);
int main() {
  int n;
  printf("\n1. Producer\n2. Consumer\n3. Exit");
  while(1) {
     printf("\nEnter your choice: ");
     scanf("%d", &n);
     switch(n) {
        case 1:
           if((mutex == 1) && (empty != 0))
             producer();
           else
             printf("Buffer is full!!\n");
           break;
        case 2:
           if((mutex == 1) && (full != 0))
              consumer();
           else
             printf("Buffer is empty!!\n");
           break;
        case 3:
           exit(0);
           break:
     }
  }
  return 0;
}
int wait(int s) {
  return (--s);
```

```
}
int signal(int s) {
  return (++s);
}
void producer() {
  mutex = wait(mutex);
  full = signal(full);
  empty = wait(empty);
  X++;
  printf("\nProducer produces the item %d\n", x);
  mutex = signal(mutex);
}
void consumer() {
  mutex = wait(mutex);
  full = wait(full);
  empty = signal(empty);
  printf("\nConsumer consumes item %d\n", x);
  X--;
  mutex = signal(mutex);
Output:
Enter the number of Producers:
Enter the number of Consumers: 1
Enter buffer capacity: 1
Successfully created producer 1
Successfully created consumer 1

    Producer

Consumer
Exit
Enter your choice: 1
Producer 1 produced 39
Buffer:39
1. Producer
Consumer
Exit
Enter your choice: 2
Consumer 1 consumed 39
Current buffer len: 0

    Producer

Consumer
Exit
Enter your choice: 3
Exiting...
Process returned 0 (0x0)
                             execution time : 15.351 s
Press any key to continue.
```

```
Produces Consumer
16/4/25
#fucludecoldio.n>
# Puchode estables in>
# Procede constitutions ins
Put nutex =1, full=0, eniphy=1, x=0;
But buffer (1);
Put wast (out s)
 Ent signal (Put s)
 void produces (Put id) [
  of (nutex == 1) el (ouply! = 0))(
nutex = wait (nutex);
       full = signal (ful);
       empty = wait(empty);
Put item = gand() 1/ 1001;
        buffer(x) = Ptom;
        point (C" Product / d productd / d \n") id, itio)
point (C" Ruffer: / d \n", puffor [x-1);
        nutex = signal (nutex);
        perentf ("Buffer to full froducer 12d wouting.")
    3 else &
 Void consumer (Ent id) C
   8f((nurtex ==1) && (full =0)) }
      nutex = wait (nutex);
       full = signal (full);
      empty = wait compty);
           item = seed() 1.100 +1;
       buffor [X) = item;
      prenty (" produced 1.d produced 1. el Val), Potter
      quant ("Buffer: Y. d In"), buffer (x-13);
```

```
best ( . In 1. beogners In 5 Consumer
       BXIH);
 Brong ( "Goter your chale")...
  guitch (choice) (
  (are 1: (nom producers so) (
    ? obel
      besorth ("No brognicer anapporprie")
    break;
 care 2: ( num consumers >0)
       communer (consumer-to);
 Pelse E
      prentf 1 100 consumers ementable")
 I break; I !
     prant ("Exiting");
     (co)tias
 default sient ( . Invalid chora!).
output:
But the no. of Produces: 1
Buter the no. of consumer: 1.
Buter the a buffer capacity:
Successfully created Produce: 1
Successfully created Consumos: 1
1. Produces
3. Gxit
Enter your choice, 1
Producer 1 produced 3 9
Buffer 39
Enter Produces
2. Consumes
3. exit
Enter your choice: 2
Consumer 1 consumed: 39
 huffer: 0.
```

```
nutex = Ergnal (nutex);
  paint f ("Bufter is ful, Producer
39 / d waterg .. In ", rd),
void consumer (ent 1d) ?
 of (cumpex == 1) && (full ! =0)) [
   nutex = weit (nuter);
    full = wait ( full)
    empty = eignal (empty);
    present ("consumer 1/d consumed 1/d In"), id, I ferry);
   Put item - buffer(x);
    pronty ("current haffel len: 1.d )" " )
   huter = signal (nutex);
   prenet ("Buffer is emply, consumer 1.d walting, id)
 Int num-producers, num-consumers, buffer capaility
 Put choice;
 Put producer -id = 1, consumer -id = 1;
 prentf ("Enter the no. of Producers"), Range ("Yd", & num producers);
 presef ("Erres the no. of Consumers")

semf ("Kd" & num-consumers)

potraf ("Enter buffer capacity:")
  econf ("Yid", Emisson - capacity);
 empty = buffer-capacity;
 Ef ( rum-producers > 0)

puents ("success fully created producer m")
  of ( rum_consumers >0)
    exent ("Euccorofally ereated product 1/11);
  Phwile(1) C
```

```
promof (" In 1. produces In 2 Consumer
        BXIT);
 Front ("Enter your choice"). . ....
  guitch (Moice) 4
  (are 1: (nom producers so) (
   ? obel
      besite ("No broances anapleable")
 Care 2:

Pf ( num consumers > 0)
       commune (consumer-to);
 Pelse E
     prentf 1 No consumers emericable").
 I break, I !
     prant ("Exiting");
     (co)tias
 default pients ( . Invalid chora!)
output:
Entor the no. of Produces !
Buter the no. of consumer: 1.
Buter the a buffer capacity:
Bucconfully created Produca: 1
Succentfully created Consumo: 1
1. Produces
3. Brit
Enter your choice 1 Produced 3 9
Buffer 39
Enter i Produces
2. Cournes
3. exit
Enter your choice: 2
Consumer 1 consumed: 39
 hufter : 0.
```

6. Write a C program to simulate the concept of Dining Philosophers problem.

```
#include <stdio.h>
#include <pthread.h>
#include <semaphore.h>
#include <unistd.h> // For usleep
#define N 5
#define THINKING 2
#define HUNGRY 1
#define EATING 0
#define LEFT (i + 4) % N
#define RIGHT (i + 1) % N
int state[N];
int phil[N] = \{0, 1, 2, 3, 4\};
sem_t mutex;
sem t S[N];
void test(int i) {
  if (state[i] == HUNGRY && state[LEFT] != EATING && state[RIGHT] != EATING) {
     state[i] = EATING;
    usleep(2000000); // Simulate eating time (2 seconds)
     printf("Philosopher %d takes fork %d and %d\n", i + 1, LEFT + 1, i + 1);
     printf("Philosopher %d is Eating\n", i + 1);
     sem_post(&S[i]);
  }
}
void take_fork(int i) {
  sem wait(&mutex);
  state[i] = HUNGRY;
  printf("Philosopher %d is Hungry\n", i + 1);
  test(i);
  sem post(&mutex);
  sem_wait(&S[i]);
  usleep(1000000); // Simulate thinking time (1 second)
}
void put fork(int i) {
  sem_wait(&mutex);
  state[i] = THINKING;
  printf("Philosopher %d putting fork %d and %d down\n", i + 1, LEFT + 1, i + 1);
```

```
printf("Philosopher %d is thinking\n", i + 1);
  test(LEFT);
  test(RIGHT);
  sem_post(&mutex);
void* philosopher(void* num) {
  while (1) {
     int^* i = num;
     usleep(1000000); // Simulate thinking before trying to eat
     take_fork(*i);
     usleep(1000000); // Simulate time spent eating
     put_fork(*i);
  }
}
int main() {
  int i;
  pthread_t thread_id[N];
  sem_init(&mutex, 0, 1);
  for (i = 0; i < N; i++) {
     sem_init(&S[i], 0, 0);
  }
  for (i = 0; i < N; i++) {
     pthread_create(&thread_id[i], NULL, philosopher, &phil[i]);
     printf("Philosopher %d is thinking\n", i + 1);
  }
  for (i = 0; i < N; i++) {
     pthread_join(thread_id[i], NULL);
  }
  return 0;
}
```

```
DiningPhilosopher.c:25:9: warming: implicit declaration of function 'sleep' [-Wimplicit-function-declaration]
     25
                         sleep(2);
 Philosopher 1 is thinking
Philosopher 2 is thinking
Philosopher 3 is thinking
Philosopher 1 is Hungry
Philosopher 2 is Hungry
Philosopher 3 is Hungr
Philosopher 3 takes fork 2 and 3
 Philosopher 3 is Eating
Philosopher 3 putting fork 2 and 3 down
Philosopher 3 is thinking
Philosopher 2 takes fork 1 and 2
 Philosopher 2 is Eating
Philosopher 3 is Hungry
Philosopher 2 putting fork 1 and 2 down
Philosopher 2 is thinking
Philosopher 1 takes fork 3 and 1
Philosopher 1 is Eating
Philosopher 2 is Hungry
Philosopher 1 putting fork 3 and 1 down
Philosopher 1 putting fork 3 and 1 down
Philosopher 1 is thinking
Philosopher 3 takes fork 2 and 3
Philosopher 3 is Eating
Philosopher 1 is Hungry
Philosopher 3 putting fork 2 and 3 down
 Philosopher 3 is thinking
Philosopher 2 takes fork 1 and 2
Philosopher 2 is Eating
Philosopher 3 is Hungry
Philosopher 2 putting fork 1 and 2 down
Philosopher 2 is thinking
```

```
void putfork (But phrum) [
                 1) energ Philosophes
                                                                       Sem-west (Emisex); [200]
Sem-west (Emisex); [200]
State [ghourn] = Thinking, hope Book 1 d and 1
 16/4/25
                                                                       state Educaria = Thinking/
printf( "Philosopher " of putting fook 1.d and /d
printf( "Philosopher " of putting fook 1.d and /d
down ", phount!, lott - 1, printing " ohning
# Proclude < Phoresod h?

# Proclude < generaphore h?

# Proclude < generaphore h?

# Proclude < constitution h?
                                                                        about " Philosopher ! d is thinking ", phrum 1)
                                                                        prints ( Philosopher ) a " test (lett);
test (lett);
test (leght);
 # defene NE
# defene trenking 2

# defene thungry |

# defene Eating 0

# defene Left (plum 4) / N
                                                                     tent (right);
                                                                      void & philosopher (word & min) &
                                                                         Put ti=nums
 # define right (phura +1) 1. Non respecting 1 to
Put state [N]: Polizish. 8;
                                                                      sicep(1))
toke-fork (*i);
elecop(0);
put-fork (*i);

Put wann (
                                           (14) 9-39-1 dot
Port philling

Sent sing.

Void tout (Ent num) L
  Pf (state [phum) == HUNGR) && State[IEFT] 1= Eating &&
                                                                      Put main() [ ] (2000+03=1[+1000
  State[EXICHT]! = Eating) [
                                                                         PHENEAD - though - Id (N); 10 1911) - ( my ) )
                                                                        But in
                                                                          80m_Mit( & mutex, 01);

80m_Mit( & mutex, 01);

801 (80=0; icN; i+1) (
 State(phone) = EATING; FORD LANGE I - LANGERY
     sleep(2);
     purentf( "enilosopher 1 d takes "fork 1 d and 1 d in",
                                                                           of Corad create (stranged id (i) (i+1));
     phum+1, left+1, phuom + D;
                                                                          for (2=0; icn; 1++) (
     porner " Philosophies tot is Gating in", pharmat).
                                                                             ptwead-join (thread-idsi), NULL)
     sem-post (Es[pmm]);
                                                                                                 9 Concer
 3
                                                                        3
 void take fork (Port phonon) ?
                                                                       Output: Philosopher 1 is thinkens
Philosopher 2 is thinkens
Philosopher a 1s thinkens
Philosopher a 1s thinkens
Philosopher 5 is thinkens
Philosopher 5 is thinkens
Philosopher 5 is thinkens
Philosopher 5 is thinkens
Philosopher 4 is thinkens
Philosopher 4 is thinkens
Philosopher 5 is thinkens
    sem_wait(&nutex);
  purint (" puriosophe "d ic Hungry", phrum+1);
test (phrum);
sem_post (& s [phrum]);
sem_wout (& s [phrum]);
    State [phaum] = HUNGASI;
steeple); proposition is
```

7. Write a C program to simulate Bankers algorithm for the purpose of deadlock avoidance.

```
#include <stdio.h>
int main() {
  int n, m, i, j, k;
  printf("Enter the number of processes: ");
  scanf("%d", &n);
  printf("Enter the number of resources: ");
  scanf("%d", &m);
  int allocation[n][m];
  int max[n][m];
  int available[m];
  int need[n][m];
  int finish[n], safeSeq[n], index = 0;
  printf("Enter the Allocation Matrix:\n");
  for (i = 0; i < n; i++)
     for (j = 0; j < m; j++)
        scanf("%d", &allocation[i][j]);
     }
  printf("Enter the MAX Matrix:\n");
  for (i = 0; i < n; i++)
     for (j = 0; j < m; j++)
        scanf("%d", &max[i][j]);
     }
  }
  printf("Enter the Available Resources:\n");
  for (i = 0; i < m; i++)
     scanf("%d", &available[i]);
  for (i = 0; i < n; i++)
     for (j = 0; j < m; j++)
        need[i][j] = max[i][j] - allocation[i][j];
     }
  for (i = 0; i < n; i++)
```

```
finish[i] = 0;
}
for (k = 0; k < n; k++)
   for (i = 0; i < n; i++)
      if (finish[i] == 0)
         int flag = 1;
        for (j = 0; j < m; j++)
            if (need[i][j] > available[j])
              flag = 0;
              break;
         }
        if (flag == 1)
            safeSeq[index++] = i;
           for (j = 0; j < m; j++)
              available[j] += allocation[i][j];
           finish[i] = 1;
        }
     }
   }
int allFinished = 1;
for (i = 0; i < n; i++)
{
   if (finish[i] == 0)
      allFinished = 0;
      break;
   }
if (allFinished)
   printf("Following is the SAFE Sequence:\n");
   for (i = 0; i < n - 1; i++)
   {
      printf("P%d -> ", safeSeq[i]);
   }
```

```
printf("P%d\n", safeSeq[n - 1]);
}
else
{
    printf("The system is NOT in a safe state.\n");
}
return 0;
}
```

```
rs.c -o Bankers } ; if ($\forall f\) { .\Bankers }
Enter number of processes and number of resources required
Enter the max matrix for all process
753
322
9 0 2
222
433
Enter number of allocated resources 5 for each process
010
200
3 0 2
211
002
Enter number of available resources
332
Resouces can be allocated to Process:2 and available resources are: 3\ 3\ 2 Resouces can be allocated to Process:4 and available resources are: 5\ 3\ 2
Resouces can be allocated to Process:5 and available resources are: 7 4 3
Resouces can be allocated to Process:1 and available resources are: 7 4 5
Resouces can be allocated to Process:3 and available resources are: 7 5 5
Need Matrix:
743
600
431
System is in safe mode
```

```
Banker & Algorithm.

#Producte < etclio. h>

# Preducte < etclio. h>

# Preducte < etclio. h>
                                                                                                          while (want < N)
                                                                                                              bool found = false;
                                                                                                               for (Aut i:0; icn; 1++) (
                                                                                                                   st (itemsura (is) ( as recorded and
 # defene war-frouses 10
# defene war-Resources 10
                                                                                                                       for (3=0;3= m;j++) [
                                                                                                                             2 (El more < (5)(3) 1
 intervency & METHOD & CONTRACTOR
                                                                                                                                 break; 114-2014 76 10 10 10 10 10 10 10
    but allow Thax : Pro unex of Now-Resources );
   gut n. m;
                                                                                                                                            013_0047 (10/01/07 - 0047_0
   Put max (max_Rounder) (max_Resources);
                                                                                                                   st a == w) (
                                                                                                                         Pay (Put k =0; lcem; k++) (
   Put avail max_ Risources ]; 3 60 - 10 48.
                                                                                                                               chaires + = alocrocks:
   Int need [max_ Procures ] [max_ ensures ].
   Put self e_sequence than _ Pro use et J.
                                                                                                                         safe sequence ( want + ) = [
                                                                                                                        formed : true;
                                                                                                             Put count =0;
    prenatingues us of bronner and anounces: ").
  prentf ("Enter & allocation matrix: "In");
                                                                                                                    provot ( "splen is not for gate state").
     bester ( come to along)
                                                                                                           Prent ("System is in saft state \n").
Prent ("Safe seguen sequence is: ").
         for (int)=0; icn; itt) [

Sout (int)=0; icn; itt)

Sout (int)=0; icn; itt)
                                                                                                             for contionien; it +> c
                                                                                                                  Sterret (" b / 19 ", 20 te - 20 dimente ())
    Pronte("Brien man matrix: \n");

Par (int i=0; icn; ita) (

for (int j=0; icm; ita) (

for (int j=0; i
                                                                                                                 pounts ("In");
     pronte ("Enter analobolable matrix ");
                                                                                                      output:
                                                                                                       Enter no. of prounes and resources: 5 3
                                                                                                       Enter allocation matrix:
      ber (604)=0) j+W) j+4) [
             Scanf ("1.d", canai (57))
                                                                                                         for (ent 1=0; icn; 1++) &
         for ( Pat ) =0 = )= m; 5+2) (
              redfiss) = max [issis - alloc (issis);
                                                                                                       Enhel max matrex
                                                                                                          4 + 3
                                                                                                          3 2 2
    902
       222
Enter available matrex:
   System is in safe no state . I was food.
 Safe sequence is: PI > P2 > P4 -> P0 -> P2
```

8. Write a C program to simulate deadlock detection

```
#include <stdio.h>
static int mark[20];
int i, j, np, nr;
int main()
{
    int alloc[10][10], request[10][10], avail[10], r[10], w[10];
    printf("\nEnter the number of processes: ");
    scanf("%d", &np);
    printf("\nEnter the number of resources: ");
    scanf("%d", &nr);
```

```
for (i = 0; i < nr; i++)
  printf("Total amount of Resource R%d: ", i + 1);
  scanf("%d", &r[i]);
printf("\nEnter the Request Matrix:\n");
for (i = 0; i < np; i++)
  for (j = 0; j < nr; j++)
  scanf("%d", &request[i][j]);
}
printf("\nEnter the Allocation Matrix:\n");
for (i = 0; i < np; i++)
  for (j = 0; j < nr; j++)
     scanf("%d", &alloc[i][j]);
for (j = 0; j < nr; j++)
  avail[j] = r[j];
  for (i = 0; i < np; i++)
     avail[j] -= alloc[i][j];
   }
for (i = 0; i < np; i++)
  int count = 0;
  for (j = 0; j < nr; j++)
     if (alloc[i][j] == 0)
        count++;
      else
         break;
  if (count == nr)
     mark[i] = 1;
for (j = 0; j < nr; j++)
  w[j] = avail[j];
for (i = 0; i < np; i++)
```

```
{
  int canBeProcessed = 0;
  if (mark[i] != 1)
     for (j = 0; j < nr; j++)
        if (request[i][j] <= w[j])</pre>
           canBeProcessed = 1;
        else {
           canBeProcessed = 0;
           break;
        }
     if (canBeProcessed)
        mark[i] = 1;
        for (j = 0; j < nr; j++)
           w[j] += alloc[i][j];
     }
  }
int deadlock = 0;
for (i = 0; i < np; i++)
  if (mark[i] != 1)
     deadlock = 1;
     break;
  }
}
if (deadlock)
  printf("\nDeadlock detected.\n");
  printf("\nNo Deadlock possible.\n");
return 0;
```

}

```
C:\Users\Admin\Documents\1wa23cs047\deadlock.exe
Enter number of processes and resources:
5 3
Enter allocation matrix:
0 1 0
2 0 0
3 0 2
2 1 1
0 0 2
Enter request matrix:
7 5 3
3 2 2
9 9 2
2 2 2
4 3 3
Enter available matrix:
3 3 2
Process 1 can finish.
Process 3 can finish.
Process 4 can finish.
System is in a deadlock state.
Process returned 0 (0x0)
                            execution time : 42.819 s
Press any key to continue.
```

```
Enter available matrex:
 8 8 2 in safe no state beyond look.
Safe sequence is: P1 > P3 > P4 > P0 > P2
                           2(+1, not ; a-1) wh
2(+2) hours? ) 11
                  and coulding land
                   to the me a fresh
            Dead lock 17 House. S. 8
                    Com fruit . spring
brent 6 ( even us of broans and sovering ).
 Score (" 1.d "/.d", &n, dm);
 scort ( 10 100, En 200); event ( m); event ( m);
Post printf("Enter allocation matrix);

for (in i=0; i <1; i+1) [

for (jeo; jen; j+1) (

Scanf(",d", halloc(; Ti)");
  prout ( " Enter reguest water x (no).
  Par (1=0; icm; it) (
Scarf ("Yid" forwill"))
   for (isopiensity) c
      boolis_zero = toute;

talizo;;cm;;t+) e

if (alloc(ix)]!=0) c
```

902

4 3 3

```
is-zero: fabe;
 break;
  fewerliteis-zero;
hoof charged; where our start is
   changed = feelse;
    for(1=0; icn; itt) (
      if ( 'finantis) c
        bool can-finishe tour;
     Par (j=0; jcm; j+4) 1
      of ( Eiglianos (1973) Sanoil (1973) C
          Can-finish = false;
  3 9 break;
      for (confinish) (
     of (con-finish) ?
       for ( = 0; ecm (e+t) (

availed + f = alloc(1)[+]

thought - brue

boint ( "Doumen I'd - "")
 thought - true 1.d on finish "?)
Sushile (changed);

beat deadlock = false;

for (i=0; i=n; i++)!

if (! finishfis)!

deadlock = true;

break;
 of (dead lock).
     parafersyrum is in a dearlook state ");
 else persent ( "anter is not en a deadlock"
```

```
Output:

Buter No. of procures and sursources (ante) /200

Finter advocation matrix:

To 3

Chres request matrix:

To 3

2 2 2

Q 0 2

2 2 2

Q 3 3

Enter anadable matrix:

Procurs I can finish
```

9.Write a C program to simulate the following contiguous memory allocation techniques a) Worst-fit

- b) Best-fit
- c) First-fit

```
#include <stdio.h>
#define MAX 25

void firstFit(int b[], int nb, int f[], int nf);
void worstFit(int b[], int nb, int f[], int nf);
void bestFit(int b[], int nb, int f[], int nf);

int main() {
    int b[MAX], f[MAX], nb, nf;
    printf("Memory Management Schemes\n");
    printf("\nEnter the number of blocks: ");
    scanf("%d", &nb);
    printf("Enter the number of files: ");
    scanf("%d", &nf);
    printf("\nEnter the size of the blocks:\n");
    for (int i = 0; i < nb; i++)
    {
        printf("Block %d: ", i + 1);
    }
}</pre>
```

```
scanf("%d", &b[i]);
  }
  printf("\nEnter the size of the files:\n");
  for (int i = 0; i < nf; i++)
  {
     printf("File %d: ", i + 1);
     scanf("%d", &f[i]);
  }
  printf("\nMemory Management Scheme - First Fit");
  firstFit(b, nb, f, nf);
  printf("\n\nMemory Management Scheme - Worst Fit");
  worstFit(b, nb, f, nf);
  printf("\n\nMemory Management Scheme - Best Fit");
  bestFit(b, nb, f, nf);
  return 0;
}
void firstFit(int b[], int nb, int f[], int nf)
{
  int bf[MAX] = \{0\}, ff[MAX] = \{0\}, frag[MAX];
  for (int i = 0; i < nf; i++)
     ff[i] = -1;
     for (int j = 0; j < nb; j++)
        if (!bf[j] \&\& b[j] >= f[i])
           ff[i] = j;
           bf[i] = 1;
           frag[i] = b[j] - f[i];
           break;
        }
     }
  }
  printf("\nFile_no:\tFile_size:\tBlock_no:\tBlock_size:\tFragment");
  for (int i = 0; i < nf; i++)
     if (ff[i] != -1)
```

```
printf("\n\%d\t\t\%d\t\t\%d\t\t\%d", i + 1, f[i], ff[i] + 1, b[ff[i]], frag[i]);
     else
         printf("\n%d\t\t%d\t\tNot Allocated", i + 1, f[i]);
  }
}
void worstFit(int b[], int nb, int f[], int nf)
  int bf[MAX] = \{0\}, ff[MAX] = \{0\}, frag[MAX];
  for (int i = 0; i < nf; i++)
     int worstldx = -1;
     for (int j = 0; j < nb; j++)
        if (!bf[j] \&\& b[j] >= f[i])
        {
           if (worstldx == -1 || b[j] - f[i] > b[worstldx] - f[i])
              worstldx = j;
        }
     ff[i] = worstldx;
     if (worstldx != -1)
         bf[worstldx] = 1;
        frag[i] = b[worstldx] - f[i];
     }
  }
  printf("\nFile_no:\tFile_size:\tBlock_no:\tBlock_size:\tFragment");
  for (int i = 0; i < nf; i++)
     if (ff[i] != -1)
        printf("\n\%d\t\t\%d\t\t\%d\t\t\%d', i + 1, f[i], ff[i] + 1, b[ff[i]], frag[i]);
        printf("\n%d\t\t%d\t\tNot Allocated", i + 1, f[i]);
  }
}
void bestFit(int b[], int nb, int f[], int nf)
{
  int bf[MAX] = \{0\}, ff[MAX] = \{0\}, frag[MAX];
  for (int i = 0; i < nf; i++)
```

```
{
   int bestldx = -1;
   for (int j = 0; j < nb; j++)
      if (!bf[j] \&\& b[j] >= f[i])
         if (bestIdx == -1 || b[j] - f[i] < b[bestIdx] - f[i])
            bestldx = j;
      }
   ff[i] = bestldx;
   if (bestIdx != -1)
      bf[bestIdx] = 1;
      frag[i] = b[bestIdx] - f[i];
   }
}
printf("\nFile_no:\tFile_size:\tBlock_no:\tBlock_size:\tFragment");
for (int i = 0; i < nf; i++)
   if (ff[i] != -1)
      printf("\n%d\t\t%d\t\t%d\t\t%d\t\t%d", i + 1, f[i], ff[i] + 1, b[ff[i]], frag[i]);
      printf("\n%d\t\t%d\t\tNot Allocated", i + 1, f[i]);
}
```

}

```
Memory Management Scheme
Enter the number of blocks: 5
Enter the number of files: 4
Enter the size of the blocks:
Block 1: 100
Block 2: 500
Block 3: 200
Block 4: 300
Block 5: 600
Enter the size of the files:
File 1: 212
File 2: 417
File 3: 112
File 4: 420

    First Fit

    Best Fit
    Worst Fit

4. Exit
Enter your choice: 1
         Memory Management Scheme û First Fit
File_no:
                                  Block_no:
                 File_size
                                                     Block_size:
234
                  212
                                   2
                                                     500
                                   5
                                                     600
                 417
                                   3
                                                     260
                  112
                 420
1. First Fit
2. Best Fit

    Worst Fit
    Exit

Enter your choice: 2
         Memory Management Scheme û Best Fit
File_no:
                 File_size
                                   Block_no:
                                                     Block_size:
1 2 3 4
                                   4
                  212
                                                     300
                 417
                                   2
                                                     500
                                   3
                                                     269
                  112
                  420
                                   5
                                                     600
```

```
4. Exit
Enter your choice: 1
       Memory Management Scheme û First Fit
File_no:
                                 Block_no:
                                                  Block_size:
                File_size
                212
                                                  500
                417
2
                                                  698
                                                  200
                112
                420
1. First Fit
Best Fit
Worst Fit
4. Exit
Enter your choice: 2
        Memory Management Scheme û Best Fit
                File_size
                            Block_no:
                                                  Block_size:
                                                  300
                417
                                                  500
                112
                                 3
                                                  200
                420
                                                  600

    First Fit

2. Best Fit
3. Worst Fit
4. Exit
Enter your choice: 3
        Memory Management Scheme û Worst Fit
                File_size
212
                                                  Block_size:
                                 Block_no:
                417
                                                  500
                112
                                                  300
                420

    First Fit

Best Fit
3. Worst Fit
  Exit
Enter your choice:
```

```
8). Wrete a c presquan to strudate poin /200
Controguous namery allocation tochologues
a) worst fet
6) Best Pet
c) terst-fot.
# Produde < state. h>
 Struct Block !
       ent 8820;
       ent decated;
1:
stout Free (
       (mt 865-8)
vold reset Blocks (struct Block bout )? it is)?
 for (that ico) icn) italy down
      blocks [13] allocated = 0;
 vord forestet (stouct Block blocks[], Put n_blocks,
 struct Fele felest), but nifeles) &
    prentf("In Memory management scheme -
    Eerst. Pet (n");
    prentf ("Freno: It Fole size It Block no: It
     Block size : \n");
     for cont 1=0; icn feles; i++) L
         fales [9]. block-no = -1;
     for ( ent j=0; 3cn-blocks; 3++) {
         ff(! blocked = 17]]. allocated & & blocks [1]. starts
         goles (7. 882e) [
```

```
filesti] block no = j+1
   block 889J. allocated = 1;
  pureficaraction place in " 6+1)
  teles [1] size ()+1, promil) . 2850).
   break;
 Pf (pelesti] block-no ==-1)[
    - perote ( "rd)+1+1-d)+1+1+1+1+1n"], 1+1,
     felessis, size);
void bost Fet ( stouch Block blocks ( ), int n-blocks,
Stout File feles [], ent n-files) [
  peant f("Int namory names. management scheme - Best 184 (n");
  perent f ("Feleno: It Felenter 21 t Block no:
  It Black - 28SE: In");
   for (ent i = 0; icn-feles; in+) [
   int bestEd x == 1;
    for ( ? ut : 30; 9 < n- blocks; 9++)
  of (1 blocks BJ. allocated && blocks BJ. 5° 22 >=
    feles [i]. spee) [
    of Chentidx == -1) 11 blocks [] . size <
    ) (asig. Exbitted lehad
           bert Idx = 1;
     135
    84 ( best Idx ! = -1) (
        blocks [best ] allocated =1;
         feles[?] block no = bentidx +1;
```

```
broad ("1.9/+/+ 1.9/+/+ 1.9
  to to d n", 1-1, pelestid size Data
   . Cosis [xb2+22d] estadd (1+ xb2+12d
     Brown & (" ). of 1 F/F / of 1414 - 141 F / "
    941, 19100 (97, 5920);
vold worst Fet (struct Block blocks (), Put noblocks
Street File Hles [] Put n-feles) &
  prent f (" In It money management schame -
   worst for (no);
  printf("Feleno: It File goe t Blockeno: It
   Block ofer: In");
   for ( fut 9:0; Kn; i+1) &
     Put woretIdx = -11.
      for ( ( nt ) = 0; SC n= $ blocks; S++) (
        "fl! blocks []]. allocated & & block s[]]. stre >=
         feles (? J. 89ca) E
         of (worstdx == -1) 11 blodes & J. size >
      glass (worsted x J. spece) 2
              Worst Idx = 1 .
    3 3
     dt (more+zgx; =-1) [
        Hocks [worst eln] allocated = (;
        geles [9] block-no = worst I dx + 1;
    Durute (" ) & to the xid the xid the 1. d la", "11"
    foles(4) Esse moretegx 11, placks (montroly).
     582e);
1. Ferst Fet
2. Bestfet
 3. Weast FET
         Navagement Schause in Best Fet
 4. Expt
            ranagement otherne It Best Fet
                          Block-no Block-size
  Faleno
                                         200
 i) FREST FRET
 2) Best Fet
 3) Worst Let
 Enter your charce: 3
   homory homogenent scheme in worst Feb
                                      Block_ 882
                                      600
                           2
                                       500
             417
             112
           420
```

```
printe (" 2 d / + 1 t / h")
 P+1, ffles [8] 8820);
hemory Management Scheme
Break the no. of blocks: 5
Buter the no. of feles: 4.
Butch the gize of the blocks:
Block 1: 100
Block 2: 400
Black 3: 200
Block 4: 300
Block 5: 600
Enter the sta of
F8101: 212
perle 2 : 617
F81e 3: 112
12/e 4:420
1) Esect Est
2) Bestret
3). Worstfit
Enter your chains !!
Memory Management Echenic à Ferst Fet
                      Block-no
         ESIQ_PEZQ
Tele_no
                       2
           212
                     5
                                    600
           417
                                    200
                     3
          (12
  8
            620
```

10.Write a C program to simulate page replacement algorithms LRU-Optimal-FIFO

#include <stdio.h>

int n, f, i, j, k; int in[100];

```
int p[50];
int hit = 0;
int pgfaultcnt = 0;
void getData() {
  printf("\nEnter length of page reference sequence: ");
  scanf("%d", &n);
  printf("\nEnter the page reference sequence: ");
  for(i = 0; i < n; i++)
     scanf("%d", &in[i]);
  printf("\nEnter number of frames: ");
  scanf("%d", &f);
}
void initialize() {
  pgfaultcnt = 0;
  for(i = 0; i < f; i++)
     p[i] = 9999;
}
int isHit(int data) {
  hit = 0;
  for(j = 0; j < f; j++) {
     if(p[j] == data) {
        hit = 1;
        break;
     }
   }
  return hit;
}
void dispPages() {
  for (k = 0; k < f; k++) {
     if(p[k]! = 9999)
        printf(" %d", p[k]);
  }
  printf("\n");
}
void dispPgFaultCnt() {
  printf("\nTotal number of page faults: %d\n", pgfaultcnt);
}
void fifo() {
  initialize();
```

```
int index = 0;
  for(i = 0; i < n; i++) {
     printf("For %d :", in[i]);
      if(isHit(in[i]) == 0) {
        p[index] = in[i];
        index = (index + 1) \% f;
        pgfaultcnt++;
        printf(" Page Fault ->");
        dispPages();
     } else {
        printf(" No page fault\n");
     }
   }
   dispPgFaultCnt();
}
void optimal() {
   initialize();
   int near[50];
   for(i = 0; i < n; i++) {
     printf("For %d :", in[i]);
     if(isHit(in[i]) == 0) {
        for(j = 0; j < f; j++) {
           int pg = p[j];
           int found = 0;
           for(k = i + 1; k < n; k++) {
              if(pg == in[k]) {
                 near[j] = k;
                 found = 1;
                 break;
              }
           if(!found)
              near[i] = 9999;
        int max = -1, repindex = -1;
        for(j = 0; j < f; j++) {
           if(near[j] > max) {
              max = near[j];
              repindex = j;
           }
        }
        p[repindex] = in[i];
        pgfaultcnt++;
        printf(" Page Fault ->");
```

```
dispPages();
     } else {
        printf(" No page fault\n");
     }
  dispPgFaultCnt();
void Iru() {
  initialize();
  int least[50];
  for(i = 0; i < n; i++) {
     printf("For %d :", in[i]);
     if(isHit(in[i]) == 0) {
        for(j = 0; j < f; j++) {
           int pg = p[j];
           int found = 0;
           for(k = i - 1; k >= 0; k--) {
              if(pg == in[k]) {
                 least[i] = k;
                 found = 1;
                 break;
              }
           if(!found)
              least[j] = -1;
        int min = 9999, repindex = -1;
        for(j = 0; j < f; j++) {
           if(least[j] < min) {</pre>
              min = least[j];
              repindex = j;
           }
        }
        p[repindex] = in[i];
        pgfaultcnt++;
        printf(" Page Fault ->");
        dispPages();
     } else {
        printf(" No page fault\n");
     }
  dispPgFaultCnt();
int main() {
```

```
int choice;
  while(1) {
     printf("\nPage Replacement Algorithms\n");
     printf("1. Enter data\n");
     printf("2. FIFO\n");
     printf("3. Optimal\n");
     printf("4. LRU\n");
     printf("5. Exit\n");
     printf("Enter your choice: ");
     scanf("%d", &choice);
     switch(choice) {
        case 1: getData(); break;
        case 2: fifo(); break;
        case 3: optimal(); break;
        case 4: Iru(); break;
        case 5: return 0;
        default: printf("Invalid choice. Try again.\n");
     }
  }
}
```

```
Enter number of pages: 15
Enter the page reference string:
7 0 1 2 0 3 0 4 2 3 0 3 1 2 0
Enter number of frames: 3
Page
        Frames
                        Page Fault
        7 - -
7 0 -
701203042303120
                Yes
                Yes
        7 0 1
                Yes
        2 0
                Yes
           ø
        2 0
                No
        2 3 1
2 3 0
                Yes
                Yes
        4
                Yes
        4 2 0
4 2 3
0 2 3
                Yes
                Yes
                Yes
          2 3 1 3
        0
                No
        0
                Yes
        0 1 2
0 1 2
                Yes
                No
Total Page Faults = 12
Process returned 0 (0x0)
                           execution time : 98.620 s
Press any key to continue.
C:\Users\Admin\Documents\1wa23cs047\Untitled1.exe
inter number of pages: 7
inter the reference string: 1 3 0 3 5 6 3
nter number of frames: 3
rames after accessing 1: 1 -
rames after accessing 3: 1 3 -
rames after accessing 0: 1 3 0
rames after accessing 3: 1 3 0
rames after accessing 5: 5 3 0
rames after accessing 6: 5 3 6
rames after accessing 3: 5 3 6
Total page faults: 5
Total page Hits: 2
Process returned 0 (0x0)
                                 execution time : 57.471 s
Press any key to continue.
C:\Users\Admin\Documents\1wa23cs047\Untitled1.exe
Enter number of pages: 7
Enter the reference string: 1 3 0 3 5 6 3
Enter number of frames: 3
Frames after accessing 1: 1
Frames after accessing 3: 1 3
Frames after accessing 0: 1 3 0
Frames after accessing 3: 1 3 0
Frames after accessing 5: 5 3 0
Frames after accessing 6: 6 3 0
Frames after accessing 3: 6 3 0
Total page faults: 5
Total page Hits: 2
Process returned 0 (0x0)
                               execution time : 19.424 s
Press any key to continue.
```

```
Lab Program 7
were a c program to stemulate
page suplacement algorithms.
a) FRFO
6) LRU
c) Opternal
#9xlude<8High>
fut maen () [
  Fut fearer, pages ( 10), n, feare (10), 1, 5, 6,
   aver!, cont=0;
   perhap (" Enter He no of pages");
   scanf ("/.d", dn);
   belief ( , Buter the book reforming think is no)
   Ra(1=0; icn; i++)
       scarf("rd", spages(iD);
   points ("From no. of ferames:");
   scarf ( "/d", eferances);
    for (9=0; ic frames ; 1+ +)
         frame [1] = -1;
    Passef(" In Page It Frame It to Page Fault (n")
     for (izo; icn; itt) ?
        avail=05
        for (1c=0; 1cc frames; 1c++) [
           Pf (frame [x] == pages (x]) [
                avail = 1;
                beeak;
         84 (avail == 0) {
            ferane (3) = pages (1);
             5 = (3+1) % frames;
             count ++;
```

```
quantf (" /d /t", pager (i));
     for (Ko; KE frames; K++) [
       Pf (frame [K] !=-1)
            prontf(ord" from (10))
             pernet (" - ");
     purnef("It yes \n");
 Selve 1
      prevette " xd " , poges (i));
      for (6-0; 100 fearnes; 10+4) [
          8f ( fram (K) != -1)
                quantf("1.d", ferane("));
               fullity("-")
        queutp ("It No m")
printf (" In Total Page Faults = 1.d ln", would)
Output:

Enter no of pages: 15

Enter the page persperence strong: 70120304230311

Enter the page persperence strong: 60120304230311

Enter no of furames: 3

Enter no of furames: 3
                                 Page Fault
           Frances
7 --
 Page
                                   yes
                                  100
             201
                                   400
                                   رعار
             420
             423
                                   401
             013
                                    Yes
               Page
 Total
```

```
LRU
# anclude < etcle . h>
2 Onson tup
 Put n, fearnes, P.S. K, faults = 0;
  prantec" Enter no of pages:");
   scamp (" 1.2", 4n);
   Put pages [n]:
  prent ("Erter the infrance string : ")
     Scanf (" 1.d", & pages [3]);
  perat("Ento mundoes of fearnes.");
Scanf ("Yed", & fearnes);
  Put frame_are [frames];
   for (Put 1-0; ic fearures; its) L
      freeze ass (i) = -1;
       + 0= [i] =0;
   for (1=0;icn;i+1)(
Put flag=0;ichianes;i+1)(
Put flag=0;ichianes;i+1)(
  gut counter=0
        Pf (franco-arrigi) = -pages(i)) E
             feage = 1;
             counter ++;
             + fune []] = counter
             break,
      P+( plac == 0) P
         but wenterme = temeco] busubor=o;
          faults ++
         for(k=1; k = frames; k+x)2
            of (Kingles Engineera) &
                men_terre = terre (K);
               men-pas=1c;
```

1

```
frame-arthur post - pages (i);
    Country +4:
    Hemo Ender por) - Courter /
  pront ( Frames after according "I.d."
  for (jeo; sc fewers ; 3+t) (
     Pf (ferane-ars [9] = = -1)
          perlot (0-11);
           perent (. /d", frame _ass (;)));
   broute " (no).
prents (" Total page foult's: 1.d \n", foult)
Put Hits = n-faults.
Quantit ( "Total page 1417. 1/d ", 140H);
Output:
Enter number of pages: 7
Enter the suference propage: 1303563
Enter number of frames : 3
Frames after accessing 1
Frames after according 3: 13-
Frames after accessing 3: 130
Frames after accessing 5: 530
Frames after accessing 6: 536
Fearus after according
                               3:536
Total page faults
 Total page Hits
```

```
Optional
# mchade < 8 toleo, h>
Put maln () (
 Put n, frames, P,3, 1c, faults = 0;
 publif (" Enter no of pages");
  scamp("1.d", In);
 Put pages this;
  priess & C. Enger ser solorours 822600 : "))
  for (1=0; 1=n; 1+1)
      Scanf("/d", & pagos (i));
   prient ("Enter ro. of pages
   gut frame anthrances)
   for (1=0; 1 c frames: 174).
    for (i=o; icn; ite) (
       Put flag=0
        for ( joo ; je frames; 1++) {
          of (frame - arr [g] = = [sugers[i]) e
              Plage = -1;
               break ;
        et (flag==0) ?
           Put pos=-1
          for (3:0; 32 fames; 3+0) L
               8 ( frame ary (3) == - 01
                 pag = 3;
           , break;
        of (60 = = -1) 5
             Put forther = P, replace fordex = 0;
```

```
fa (7:0; 3 c frames; 3+1) (
  Put found = 0.
 for ( 12- Pt1 ; Ken ; K++) (
      of (francon (3) == pages Ex3) (
          Pf (k > farthest) (
            farthest = K;
             replace - ander = 3;
     found = 1;
     break;
 3
  Pr (found) (
     replace and ex = 1;
     break,
 pos= supla co_Prodex)
frame arr [pos] = pages [].
prents ( " teams after acossing . I.d: ", pages [+]);
for (j=0; )x frames; (g) == -1)
        printf("-");
         bisnot (a.190, brane - orself)).
poster ("Total page faulty: Yed In", faults);
Put 444 5, = n-faults;
prenty ( "Total page Hits: "/.d ", "Hits );
```

```
output:
giver number of pages: 7
Enter the reference strong:
1303563
       number of frames: 3
Entel
             accenting 1 : 1--
Frames after
            accenting
                      3: 13-
Frames after
                      0:130
Frances after accessing
Frances after accossing
              accessing 5:
                           530
Francis after
Frames after according 6:
                           630
Former after according ?:
Total page faults
Total page
```

Start