VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.



LAB REPORT on

OPERATING SYSTEMS

Submitted by

VANSHIKA KATARIA(1WA23CS031)

in partial fulfillment for the award of the degree of BACHELOR OF ENGINEERING in COMPUTER SCIENCE AND ENGINEERING



B.M.S. COLLEGE OF ENGINEERING (Autonomous Institution under VTU) BENGALURU-560019 Feb-2025 to 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 – 23CS4PCOPS" carried out by Vanshika kataria(1WA23CS031), who is Bonafide student of B. M. S. College of Engineering. It is in partial fulfilment for the award of Bachelor of Engineering in Computer Science and Engineering of the Visvesvaraya Technological University, Belgaum during the year Feb 2025- June 2025. The Lab report has been approved as it satisfies the academic requirements in respect of a OPERATING SYSTEMS - (23CS4PCOPS) work prescribed for the said degree.

Seema Patil Assistant Professor Department of CSE BMSCE, Bengaluru Dr. Kavitha Sooda Professor and Head Department of CSE BMSCE, Bengaluru

Sl. No.	Experiment Title	Page No.
1.	Write a C program to simulate the following non-pre-emptive CPU scheduling algorithm to find turnaround time and waiting time. →FCFS → SJF (pre-emptive & Non-preemptive)	1-10
2.	Write a C program to simulate the following CPU scheduling algorithm to find turnaround time and waiting time. → Priority (pre-emptive & Non-pre-emptive) →Round Robin (Experiment with different quantum sizes for RR algorithm)	11-19
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.	20-24
4.	Write a C program to simulate Real-Time CPU Scheduling algorithms: a) Rate- Monotonic b) Earliest-deadline First c) Proportional scheduling	24-34
5.	Write a C program to simulate producer-consumer problem using semaphores	35-38
6.	Write a C program to simulate the concept of Dining Philosophers problem.	39-43
7.	Write a C program to simulate Bankers algorithm for the purpose of deadlock avoidance.	44-47
8.	Write a C program to simulate deadlock detection	48-51
9.	Write a C program to simulate the following contiguous memory allocation techniques a) Worst-fit b) Best-fit c) First-fit	52-56
10.	Write a C program to simulate page replacement algorithms a) FIFO b) LRU c) Optimal	57-63

Course Outcomes

C01	Apply the different concepts and functionalities of Operating System
C02	Analyse various Operating system strategies and techniques
C03	Demonstrate the different functionalities of Operating System.
C04	Conduct practical experiments to implement the functionalities of Operating system.

Program -1

Question:

Write a C program to simulate the following non-pre-emptive CPU scheduling algorithm to find turnaround time and waiting time.

```
→FCFS
→ SJF (pre-emptive & Non-preemptive)
Code:
=>FCFS:
#include <stdio.h>
void calculateTimes(int processes[], int n, int at[], int bt[], int ct[], int tat[], int wt[]) {
  int completion = 0;
  for (int i = 0; i < n; i++) {
     if (completion < at[i]) {
        completion = at[i];
     }
     completion += bt[i];
     ct[i] = completion;
     tat[i] = ct[i] - at[i];
     wt[i]=tat[i]-bt[i];
  }
}
void displayResults(int processes[], int n, int at[], int bt[], int ct[], int tat[], int wt[]) {
  float total tat = 0, total wt = 0;
  printf("\nProcess\tAT\tBT\tCT\tTAT\tWT\n");
  for (int i = 0; i < n; i++) {
     total tat += tat[i];
     total wt += wt[i];
     printf("%d\t%d\t%d\t%d\t%d\t%d\n", processes[i], at[i], bt[i], ct[i], tat[i], wt[i]);
```

```
}
  printf("\nAverage Turnaround Time = %.2f", total_tat / n);
  printf("\nAverage Waiting Time = %.2f\n", total_wt / n);
}
int main() {
  int n;
  printf("Enter no. of processes : ");
  scanf("%d",&n);
  int processes[n];
  for(int i=0; i< n; i++){
     processes[i]=i+1;
       }
  int at[n];
  for(int i=0;i<n;i++){
     printf("Enter arrival time of process %d : ",processes[i]);
     scanf("%d",&at[i]);
       }
  int bt[n];
  for(int i=0;i<n;i++){
     printf("Enter burst time of process %d : ",processes[i]);
     scanf("%d",&bt[i]);
       }
  int ct[n], tat[n], wt[n];
  calculateTimes(processes, n, at, bt, ct, tat, wt);
  displayResults(processes, n, at, bt, ct, tat, wt);
  return 0;
}
```

```
Process returned 0 (0x0) execution time : 103.967 s
Press any key to continue.
                              int ct[n], tat[n], wt[n];
                              calculateTimes(processes, n, at, bt, ct, tat, wt);
displayResults(processes, n, at, bt, ct, tat, wt);
                              return 0:
                                                               Prope 1
             CAB-1.
          pollowing Non- prentive CN scheduling algorithms to final treir tornounced time and uniting time.

ECFS (first come first serve)

SJ-F (Shortot Job Wathing V.P. & Prency)
                                                       (CT-AT) (TAT-B)
           PROCESS AT BT CT THE WT &
                                     3 10 10
           Pu 0 6 20 20 14

Pu 0 6 20 20 14

$ 12.75 $ = 2.75
             Gautt chart
          # include statio h>
void calculateTimes (int procoses[], int n,
not at [], got bt[], int ct[], int tat[],
tot wt[]) {
               for (int of 1/0); 2(0); 2(1) &
```

```
completion at [i]:

[ completion to be beli];

[ completion to believe to beli
```

=>SJF(Non-preemptive):

```
#include <stdlib.h>
#include <stdlib.h>
struct Process {
    int id;
    int bt;
    int at;
    int ct;
    int tat;
    int wt;
};

int compareArrivalTime(const void *a, const void *b) {
    return ((struct Process*)a)->at - ((struct Process*)b)->at;
}

void calculateTimes(struct Process processes[], int n) {
    int time = 0;
```

```
int completed = 0;
  while (completed \leq n) {
     int shortest = -1;
     int min burst = 1000000;
     for (int i = 0; i < n; i++) {
       if (processes[i].at <= time && processes[i].ct == 0) {
          if (processes[i].bt < min burst) {
             min burst = processes[i].bt;
             shortest = i;
     if (shortest == -1) {
       time++;
     } else {
       processes[shortest].ct = time + processes[shortest].bt;
       processes[shortest].tat = processes[shortest].ct - processes[shortest].at;
       processes[shortest].wt = processes[shortest].tat - processes[shortest].bt;
       time = processes[shortest].ct;
       completed++;
     }
  }
void calculateAvg(struct Process processes[], int n) {
  int total wt = 0, total tat = 0;
  for (int i = 0; i < n; i++) {
     total wt += processes[i].wt;
     total tat += processes[i].tat;
  printf("\navg wt = \%.2f", (float)total wt / n);
  printf("\navg tat = \%.2f", (float)total tat / n);
```

```
}
int main() {
  int n;
  printf("no. of processes: ");
  scanf("%d", &n);
  struct Process processes[n];
  printf("enter bt and at: \n");
  for (int i = 0; i < n; i++) {
     processes[i].id = i + 1;
     printf("bt %d: ", i + 1);
     scanf("%d", &processes[i].bt);
    printf("at %d: ", i + 1);
     scanf("%d", &processes[i].at);
     processes[i].ct = 0;
  }
  qsort(processes, n, sizeof(struct Process), compareArrivalTime);
  calculateTimes(processes, n);
  calculateAvg(processes, n);
  return 0;
}
Result:
no. of processes: 4
enter bt and at: bt 1: 7
at 1: 0
bt 2: 3
at 2: 8
bt 3: 4
at 3: 3
bt 4: 6
at 4: 5
avg wt = 4.00
avg tat = 9.00%
```

ST Non-how.	-
	3
# Include astalon h >	
# include (stallihe A?	1/ (0)
	if (sharket == +1) E
struct Process ?	
intid:	STORY OF THE PARTY AND ADDRESS OF THE PARTY AN
int bt : 1 and 1 and 1 and 1	they es (shortest) of a timet a
ent ati	frames (shorted) at a time t power process (shorted) at a process (s
int ct;	tat = pocone [clasted]
Intitot:	Parage Color Luca Proconce School of
int wt:	Process (Shrahelf) at process (Secretal) at process (Shrahelf) at process (Shortelf) at
3;	time = processes [ch = processes should be
Colon Colon Colon Data La	time = procones [shorter]. et
int compare Arrival Time (court void a continu	1 minute Lt.
Edum (1 Street fraces 7) a) - at 11 Street to	and of the state of
3 (Street for	A Survey of the
3	int total and struct process processed int to the
	inct fracu processes]
hid calculate Time (court void *a, and void *)	int total wit = a, total test = a;
int time = 0;	
int completed = 0;	for (int i= 0; (2n; i+t)) = total wt + = procenu [i], wt;
whole (impleted (n) ?	total wit + = macroner (5)
hat shortest = - 1;	total - tat += processes (2) -tat;
Int mi	\$ \$700.000m (1.7. Tal ; -
In+ min_bout = 1000000;	
Jay (1) 1	print ("wt = x. a ;", (foot) to tal wt/n);
21/2 (Cn; f+) 8	print ("tat = 16 8 1" ("West) total + 1/1) 1
fax (hat i=o; i(x ; jet) & I (paceno [i) at (= time)]	I de la description de la constitución de la consti
11 procen [i] st == 0) 8	A I I I
from (i) (+= 0) }	int main() ?
	int n:
shortest of ?	pantl (no of process:)
32	most ("ha", en);
	the state of the s

	4	
	Anust freen procones (n);	
4		
	pant (" enter bt and at ") :	
	fint) (" enter of and for (Int i = 0.) is n : (++) 3	
	process [19] 1 d = 11;	
	pant (" of 1.8 : 11, ?+1);	
the better	(" " d", 2 processes (1), 14)	
	pild ("at "d:", [+1);	
	scar (1.d , & procond (1) at):	
	process (P) . ct = 0;	
	S to hat digneral	
	good (processes, n, size of (Struct proces), compare minuto	
	Comparaning	1
- Silvery		400
	Calabate Times (processes, n);	-
	calculate Avg (processes, n);	-
	petum a; (fracenes, n);	-
		-
	Silverson and the Details	-
124		-
	Output:	-
		-
	No. of fincers a 2 4	-
	Coton Lt - 1	-
	cuter It and at : 1	
	at 1:0	
	# 3 3	
	at 2 : 6	
	bt 3 : 4	1
	. 0	
	at 4:6 avg tat = 9.00%	
	1.0	

=>SJF(Preemptive):

#include <stdio.h>

```
#include imits.h>
struct Process {
  int id;
  int bt;
  int at;
  int rt;
  int ct;
  int tat;
  int wt;
};
void calculateTimes(struct Process processes[], int n) {
  int completed = 0, time = 0, shortest = -1;
  int min burst = INT MAX;
  while (completed \leq n) {
     shortest = -1;
     min_burst = INT_MAX;
     for (int i = 0; i < n; i++) {
        if (processes[i].at <= time && processes[i].rt > 0) {
          if (processes[i].rt < min burst) {</pre>
             min burst = processes[i].rt;
             shortest = i;
     if (shortest == -1) {
       time++;
        continue;
```

```
}
     processes[shortest].rt--;
     time++;
     if (processes[shortest].rt == 0) {
       completed++;
       processes[shortest].ct = time;
       processes[shortest].tat = processes[shortest].ct - processes[shortest].at;
       processes[shortest].wt = processes[shortest].tat - processes[shortest].bt;
     }
  }
void calculateAvg(struct Process processes[], int n) {
  int total wt = 0, total tat = 0;
  for (int i = 0; i < n; i++) {
     total wt += processes[i].wt;
     total tat += processes[i].tat;
  }
  printf("\nAvgWT = \%.2f", (float)total wt / n);
  printf("\nAvg TAT = %.2f", (float)total tat / n);
}
int main() {
  int n;
  printf("No. of processes: ");
  scanf("%d", &n);
  struct Process processes[n];
  printf("Enter BT and AT: \n");
  for (int i = 0; i < n; i++) {
```

```
processes[i].id = i + 1;
printf("BT %d: ", i + 1);
scanf("%d", &processes[i].bt);
printf("AT %d: ", i + 1);
scanf("%d", &processes[i].at);
processes[i].rt = processes[i].bt;
}
calculateTimes(processes, n);
calculateAvg(processes, n);
return 0;
}
```

```
No. of processes: 4
Enter BT and AT:
BT 1: 8
AT 1: 0
BT 2: 4
AT 2: 1
BT 3: 9
AT 3: 2
BT 4: 5
AT 4: 3

Avg WT = 6.50
Avg TAT = 13.00%
```

```
# include < Stdia h >

# include < Ilmiti h

Atrick Brocers &

Int id;

Int bt;

Int tat;

Int tat;

Int tot;

Int tot;

Int coupleted = a, time = a, shortet = 1;

Int min bust = Thir, KAX;

Wille (completed < n) &

Sharlest = -1;

min bust = Thir, KAX;

If (procone (1), at < = time se

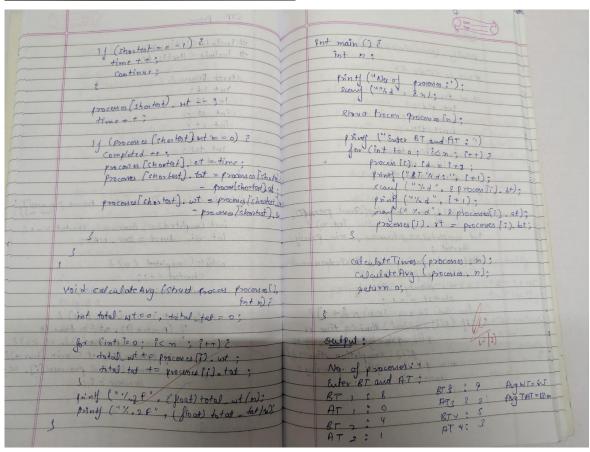
(procone (1), et < emin bust)

If (procone (1), et < emin bust)

If (procone (1), et < emin bust)

Mostet = 1;

Shortest = 1;
```



Program-2

}

Question

Write a C program to simulate the following CPU scheduling algorithm to find turnaround time and waiting time.

```
and waiting time.
→ Priority (pre-emptive & Non-pre-emptive)
→Round Robin (Experiment with different quantum sizes for RR algorithm)
=>PRIORITY(pre-emptive)
#include <stdio.h>
struct Process {
  int id, arrival time, burst time, priority, completion time, turnaround time,
waiting time;
};
void swap(struct Process *a, struct Process *b) {
  struct Process temp = *a;
  *a = *b;
  *b = temp;
}
void sort by priority(struct Process p[], int n) {
  for (int i = 0; i < n - 1; i++) {
     for (int j = 0; j < n - i - 1; j++) {
        if (p[j].priority > p[j + 1].priority ||
         (p[j].priority == p[j + 1].priority && p[j].arrival time > p[j +
1].arrival time)) {
          swap(&p[j], &p[j+1]);
```

```
}
}
int main() {
  int n;
  printf("Enter the number of processes: ");
  scanf("%d", &n);
  struct Process p[n];
  printf("Enter Arrival Time, Burst Time, and Priority for each process:\n");
  for (int i = 0; i < n; i++) {
    p[i].id = i + 1;
    scanf("%d %d %d", &p[i].arrival time, &p[i].burst time, &p[i].priority);
  }
  sort by priority(p, n);
  int current time = 0;
  float total tat = 0, total wt = 0;
  for (int i = 0; i < n; i++) {
    if (current time < p[i].arrival time) {
       current time = p[i].arrival time;
     }
     p[i].completion time = current time + p[i].burst time;
     p[i].turnaround time = p[i].completion time - p[i].arrival time;
     p[i].waiting time = p[i].turnaround time - p[i].burst time;
    current time = p[i].completion time;
    total tat += p[i].turnaround time;
    total wt += p[i].waiting time;
  }
  printf("\nProcess AT
                                  Priority
                           BT
                                               CT
                                                      TAT WT\n'');
```

```
for (int i = 0; i < n; i++) {
    printf("P%d %d %d %d %d %d %d %d\n", p[i].id,
p[i].arrival_time, p[i].burst_time, p[i].priority, p[i].completion_time,
p[i].turnaround_time, p[i].waiting_time);
}
printf("\nAverage Turnaround Time: %.2f\n", total_tat / n);
printf("\nAverage Waiting Time: %.2f\n", total_wt / n);
return 0;
}</pre>
```

```
Enter the number of processes: 4
Enter Arrival Time, Burst Time, and Priority for each process:
0 10 3
0 1 1
3 2 3
5 1 4
Process AT
                 вт
                         Priority
                                          CT
                                                   TAT
                                                           WT
        0
                                                           0
Р1
                 10
        0
                         3
                                          11
                                                   11
Р3
                         3
                                                           8
        3
                 2
                                          13
                                                   10
Р4
                 1
                         4
        5
                                          14
                                                   9
                                                           8
Average Turnaround Time: 7.75
Average Waiting Time: 4.25
Process returned 0 (0x0)
                             execution time : 37.441 s
Press any key to continue.
```

	Q total
	if (shortest == 1) F time + +: continue;
Blooty (Ben)	time is
	Comb.
# include & stdio. A >	2 Primue,
Struct Process &	
int pla	p[shortest]
womal time; burst time;	p (shortest). seriamia - time
byrst time;	O f chastes ty . Hemain a 1
priority 3	estimpleted ++, James = 0)
remaining that	p(shortest), turnarand, there of shorter completion time - Pishorter), and
completion time!	CONTROL TIME TO THE
waiting time!	proximation time
the naround time;	turnaround time - plehostest burst time total waiting + = plehostest burst time
	total waiting + = p(sharper)
void priority Premytive (Struct Boxen pl) in	total waiting of plehostest) bust to fotal tornaround += Plehostest J. waiting
int time = 0; min privity shortest	total tornaround + - Plinestot J. turn-
completed = 0;	3 around tive;
float total waiting = 0	3
total tron around to;	time + +;
we was a result	3 · la Daidy may be
(or (int i = 0; ixn; i+7)	3
p [i] - genaining time = p (i), bloost the	· Print (& Press Paraile and 1 Asin 2
while (completed < n) &	act ("DIB) 1 1 7 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1
	Printf ("Frem. Brighty Scheduling) Printf ("PID) + HT) + BT) + WY + CT) +
min_priorty = 9999;	
8hortst = -1 3	for(in/[=0; (<n; ?++)<="" td=""></n;>
for (1mt 1=0; ((m; (++)))	print ("7.21+",d1+",d1+",d1+",d1+",d1+",d1+",d1+",d
if f(1) aveiral time (= time;	P(i) pid)
82 Mis. genoining time > 0 22	p Pi). avuival. time, p (i) buse
min Priority - of it private	ofi's entirete.
min knowly = plid. privally;	exant (" Avg. waiting time: 1. 24 /
1	print type walling total it
1	tatal-waitin

	15
	Ober 1
Alto of OTT.	L sax C
print ("Aug Tinnaramott ne 15 2 P los)	Output.
total Donason m 3	a constitution of the state of
int main () F	Enter no of process: 7
int main () P	Enter Process of D. AT. RT. Q.
Print (" E to a of a state of	Enter Process TD, AT, BT, Privity: 1083
Print ("Enter no of processes"),	3324
	1 - 14 0 15
0 10	25 5 6 2
for (int 1003 (< m : 6++) 8	:6656
erent ("Entr forces I D Arrive DT	2 10 11
for (int i= 0; (< m; &+ +) & for (int i= 0; (< m; &+ +) & printy ("Entr Procen I D, Assiraf Time, Brost Time Privaty") 1 of 1 d 1 d 1 d 1 d 2 d 2 d 2 d 2 d 2 d 2 d	0 10 0 11 21
soul (2) d 1. d 2 d 2 20 d 2 0 0 181	Premptive Prierity Scheduling
2 pli). avairal thre, pli). boost time, 2 pli). priority);	A MARKET COMPANY OF THE PARTY O
p (1) . borst time.	THE WI
2 Phis priority):	2 1 2 4 15 16 14
3	2 2 4 4 21 16 14
1 to a facility	y y 1 5 29 18 12
priority freughise (p, n);	5 (2 19 4 1
edun 03	6 6 5 6 29 01 15
Court of the Court	7 10 (1) 11 0
04/+2+1/2+/+04/+04/+06/ BIO'S 12-6	and I want to the
1/3/4/2	Ang WT 5 9.86
7.00 - 3.0	Avg TAT : 18.74
The state of the s	1100
	and the same of the same first the same of
The Array of July 18 18 18 18 18 18 18 18 18 18 18 18 18	The state of the s
A STATE OF THE STA	t work tot

```
=>PRIORITY(Non-pre-emptive)
#include <stdio.h>
struct Process {
  int id, arrival time, burst time, priority, completion time, turnaround time,
waiting time;
};
void swap(struct Process *a, struct Process *b) {
  struct Process temp = *a;
  *a = *b;
   *b = temp;
}
void sort by priority(struct Process p[], int n) {
  for (int i = 0; i < n - 1; i++) {
     for (int j = 0; j < n - i - 1; j++) {
        if (p[j].priority > p[j + 1].priority ||
          (p[j].priority == p[j + 1].priority && p[j].arrival time > p[j + 1].priority && p[j].arrival time > p[j + 1].priority
1].arrival_time)) {
           swap(&p[j], &p[j+1]);
int main() {
```

```
int n;
  printf("Enter the number of processes: ");
  scanf("%d", &n);
  struct Process p[n];
  printf("Enter Arrival Time, Burst Time, and Priority for each process:\n");
  for (int i = 0; i < n; i++) {
     p[i].id = i + 1;
     scanf("%d %d %d", &p[i].arrival time, &p[i].burst time, &p[i].priority);
  }
  sort by priority(p, n);
  int current time = 0;
  float total tat = 0, total wt = 0;
  for (int i = 0; i < n; i++) {
     if (current time < p[i].arrival time) {
       current time = p[i].arrival time;
     }
     p[i].completion time = current time + p[i].burst time;
    p[i].turnaround time = p[i].completion time - p[i].arrival time;
     p[i].waiting time = p[i].turnaround time - p[i].burst time;
     current time = p[i].completion time;
    total tat += p[i].turnaround time;
     total wt += p[i].waiting time;
  }
  printf("\nProcess\tAT\tBT\tPriority\tCT\tTAT\tWT\n");
  for (int i = 0; i < n; i++) {
     printf("P%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\n", p[i].id, p[i].arrival time,
p[i].burst time, p[i].priority, p[i].completion time, p[i].turnaround time,
p[i].waiting time);
  }
```

```
printf("\nAverage Turnaround Time: %.2f", total_tat / n);
printf("\nAverage Waiting Time: %.2f\n", total_wt / n);
return 0;
}
```

```
Enter the number of processes: 4
Enter Arrival Time, Burst Time, and Priority for each process:
0 5 4
2 4 2
2 2 6
4 4 3
Process AT
                     вт
                               Priority
                                                   \mathsf{CT}
                                                              TAT
                                                                        WT
                               2
P2
           2
                     4
                                                   6
                                                              4
                                                                        0
Р4
                                                   10
                               4
6
Ρ1
          0
                     5
                                                   15
                                                              15
                                                                        10
Р3
                                                   17
                                                                        13
                                                              15
Average Turnaround Time: 10.00
Average Waiting Time: 6.25
Process returned 0 (0x0)
                                   execution time : 27.552 s
Press any key to continue.
```

	Lab-2.	Orte
		R mm C
	Priority - (Non - Presen Price) ion to	el (Shartest = - 10
	Thomas Complete	(Shortest = = -1) { time 4 -t ;
	# include (Stdio. B)	Cartin
	The source of pix (1) (\$1.5)	continue;
	Struct Process &	e [shortert] a
	int pid:	P(shortest), completion three = time +
	assivel time:	P(shostor) busst time = time +
	bunch than	P[Shortest] . fun a arount time= P[shorted].
	priority of	completion of shortest) animal time
	Priority;	p (shortest). Walting time = p (shostest).
	Completion time:	p (shortest) waiting time = p [shortest]. there or our time - p [shortest], burst thee;
	furn acound time ;	
	l. Time;	total waiting to flohortoot) witing in
	(1-13 " p. N") 11-29	
	Void Print 1 40 B 1 11 Ca 1	total ternamound to p (shortest).
290 (5)	Void Priority Non-Procuptive (Street proceed),	turna round time
	1nt cond lile int n) ?	time + = plshortest) burst - time;
	Int completed = 0, time = 0, min priority.	completed + + &
		34 dilla spelet trad
	float total waiting = 0; total ternarant-o	print ("In Non- preemptive priority
	while (completed (n) 5	scheduling ? \n');
	min priority = 999:	exist ("PID) AAT (+87) + ET (+TA)
		print ("n Non- freemptive priority "cheduling: \n"); ("PID \ ATT \ +BT \ +TT \ \ +TT \ \ \ \ \ \ \ \ \ \ \ \
	for (Int 7 = 0; E(n; i++) E	
	if (Pli) ovival time <= time es	10 /47/ 1 / 1 / 1 / 1 / 1
	p[i] . Priority < min priority 88	
	P[I] Completion - time = 0) ?	plid plid , plid , plid , privily , privily , plid , privily , plid , privily , privil
	min priority = PFil. priority;	
	shortat= 173	to the third decourt
- Dat His		grid owniting timel;
		411

```
print ("himage waiting time is

print) ("himage Turnamund ethics:

print) ("Areage Turnamund ethics:

print) ("Areage Turnamund ethics:

print) ("Areage Turnamund ethics:

print) ("Enter number of process")

print) ("Enter number of process")

print ("Enter number of process")

print ("Ind" + En);

Non Freemptine Briefly Strawling

struct process f(n);

print ("Enter number in print);

print ("Ind" + En);

PID AT BT P CT FAT WT

for (Int is a it n'; i++) if

print ("Enter number in print);

print ("Ind" + En);

Print ("Ind" + En
```

=>ROUND ROBIN

```
done = 0;
          if (processes[i].rt > quantum) {
             time += quantum;
             processes[i].rt -= quantum;
          } else {
             time += processes[i].rt;
             processes[i].ct = time;
             processes[i].tat = processes[i].ct - processes[i].at;
             processes[i].wt = processes[i].tat - processes[i].bt;
             processes[i].rt = 0;
             completed++;
     if (done) time++;
}
void calculateAvg(struct Process processes[], int n) {
  int total wt = 0, total tat = 0;
  for (int i = 0; i < n; i++) {
     total wt += processes[i].wt;
     total tat += processes[i].tat;
  }
  printf("\nAvg WT = \%.2f", (float)total wt / n);
  printf("\nAvg TAT = \%.2f", (float)total tat / n);
}
int main() {
```

```
int n, quantum;
  printf("No. of processes: ");
  scanf("%d", &n);
  struct Process processes[n];
  for (int i = 0; i < n; i++) {
    processes[i].id = i + 1;
    printf("BT %d: ", i + 1);
    scanf("%d", &processes[i].bt);
    printf("AT %d: ", i + 1);
    scanf("%d", &processes[i].at);
    processes[i].rt = processes[i].bt;
  }
  printf("Time Quantum: ");
  scanf("%d", &quantum);
  roundRobin(processes, n, quantum);
  calculateAvg(processes, n);
  return 0;
}
```

```
No. of processes: 5
BT 1: 8
AT 1: 0
BT 2: 2
AT 2: 5
BT 3: 7
AT 3: 1
BT 4: 3
AT 4: 6
BT 5: 5
AT 5: 8
Time Quantum: 3

Avg WT = 10.40
Avg TAT = 15.40
```

Program-3

Question:

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.

```
=>MULTILEVEL QUEUE
#include <stdio.h>
#define MAX PROCESSES 10
#define TIME QUANTUM 2
typedef struct {
  int id;
  int burst time;
  int arrival time;
  int queue; // 1 for system process (RR), 2 for user process (FCFS)
  int remaining time;
  int waiting time;
  int turnaround time;
  int response time;
} Process;
void round robin(Process processes[], int n, int tq, int *time) {
  int done, i;
  do {
    done = 1;
    for (i = 0; i < n; i++)
```

```
if (processes[i].queue == 1 && processes[i].remaining time > 0) {
          done = 0;
          if (processes[i].remaining time == processes[i].burst time)
            processes[i].response time = *time - processes[i].arrival time;
          if (processes[i].remaining_time > tq) {
            *time += tq;
            processes[i].remaining time -= tq;
          } else {
            *time += processes[i].remaining_time;
            processes[i].waiting time = *time - processes[i].burst time -
processes[i].arrival time;
            processes[i].turnaround time = *time - processes[i].arrival time;
            processes[i].remaining time = 0;
  } while (!done);
}
void fcfs(Process processes[], int n, int *time) {
  for (int i = 0; i < n; i++) {
    if (processes[i].queue == 2) {
       if (*time < processes[i].arrival_time)
          *time = processes[i].arrival time;
       processes[i].response time = *time - processes[i].arrival time;
       processes[i].waiting time = *time - processes[i].arrival time;
       *time += processes[i].burst time;
       processes[i].turnaround time = *time - processes[i].arrival time;
     }
```

```
}
}
void calculate average(Process processes[], int n) {
  float total waiting = 0, total turnaround = 0, total response = 0;
  for (int i = 0; i < n; i++) {
     total waiting += processes[i].waiting time;
    total turnaround += processes[i].turnaround time;
    total response += processes[i].response time;
  printf("\nAverage Waiting Time: %.2f", total waiting / n);
  printf("\nAverage Turn Around Time: %.2f", total turnaround / n);
  printf("\nAverage Response Time: %.2f", total response / n);
  printf("\nThroughput: %.2f\n", n / (total turnaround / n));
}
int main() {
  int n, time = 0;
  Process processes[MAX PROCESSES];
  printf("Enter number of processes: ");
  scanf("%d", &n);
  printf("Queue 1 is system process\nQueue 2 is User Process\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);
    processes[i].id = i + 1;
    processes[i].remaining time = processes[i].burst time;
    processes[i].waiting time = 0;
```

```
processes[i].turnaround_time = 0;
processes[i].response_time = 0;
}
round_robin(processes, n, TIME_QUANTUM, &time);
fcfs(processes, n, &time);
printf("\nProcess\tWaiting Time\tTurn Around Time\tResponse Time\n");
for (int i = 0; i < n; i++) {
    printf("\%d\t\%d\t\t\%d\t\t\%d\n", processes[i].id, processes[i].waiting_time, processes[i].turnaround_time, processes[i].response_time);
}
calculate_average(processes, n);
return 0;
}</pre>
```

```
Enter number of processes: 4
Queue 1 is system process
Queue 2 is User Process
Enter Burst Time, Arrival Time and Queue of P1: 2 0 1
Enter Burst Time, Arrival Time and Queue of P2: 1 0 2
Enter Burst Time, Arrival Time and Queue of P3: 5 0 1
Enter Burst Time, Arrival Time and Queue of P4: 3 0 2
Process Waiting Time
                        Turn Around Time
                                                 Response Time
        0
        7
                        8
                                                 7
3
        2
                        7
                                                 2
                        11
        8
Average Waiting Time: 4.25
Average Turn Around Time: 7.00
Average Response Time: 4.25
Throughput: 0.57
Process returned 0 (0x0)
                           execution time: 71.241 s
```

in the system processes and were processes agreem frocesses are to be given higher processes are to be given higher than user processes. as the processes in each queue. Solut. It include a station is a three processes in each queue. It define MAX processes to the define TIME - AVANTUM a type def struct ? Int burst time; Int which there; Int queue; Int remaining time;	if (processes [1]. remaining three = processed] processes [i]. response three = time = 1] (processes [i]. autival time; three t= to; processes [i]. remaining = to; three t= processes [i]. remaining = to; selse ? * time t= processes [i]. remaining = to; processes [i]. remaining = time = processed avisoration = processed avisoration = time = time = processes [i]. transaction time = time = processes [i]. remaining time = 0;
in the system processes and were processes agreem frocesses are to be given higher processes are to be given higher than user processes. as the processes in each queue. Solut. It include a station is a three processes in each queue. It define MAX processes to the define TIME - AVANTUM a type def struct ? Int burst time; Int which there; Int queue; Int remaining time;	processe [i] austral time; the total remaining time 74) & processes [i] remaining time; processes [i] remaining time; processes [i] remaining time; processes [i] hasting time = time - processed processes [i] turnaround time = time - process [i] remaining time = 0;
in the system processes and were processes agreem frocesses are to be given higher processes are to be given higher than user processes. as the processes in each queue. Solut. It include a station is a three processes in each queue. It define MAX processes to the define TIME - AVANTUM a type def struct ? Int burst time; Int which there; Int queue; Int remaining time;	processe [i] austral time; the total remaining time 74) & processes [i] remaining time; processes [i] remaining time; processes [i] remaining time; processes [i] hasting time = time - processed processes [i] turnaround time = time - process [i] remaining time = 0;
in the system processes and were processes agreem frocesses are to be given higher processes are to be given higher than user processes. as the processes in each queue. Solut. It include a station is a three processes in each queue. It define MAX processes to the define TIME - AVANTUM a type def struct ? Int burst time; Int which there; Int queue; Int remaining time;	processe [i] austral time; the total remaining time 74) & processes [i] remaining time; processes [i] remaining time; processes [i] remaining time; processes [i] hasting time = time - processed processes [i] turnaround time = time - process [i] remaining time = 0;
in the system processes and were processes agreem frocesses are to be given higher processes are to be given higher than user processes. as the processes in each queue. Solut. It include a station is a three processes in each queue. It define MAX processes to the define TIME - AVANTUM a type def struct ? Int burst time; Int which there; Int queue; Int remaining time;	processe [i] austral time; the total remaining time 74) & processes [i] remaining time; processes [i] remaining time; processes [i] remaining time; processes [i] hasting time = time - processed processes [i] turnaround time = time - process [i] remaining time = 0;
solur processes are to be given higher processes are to be given higher processes are processes are processes are processes are processes in each queue. Solur. It include < st dio : 6 ? It define MAX processes to It define TIME - QUANTUM a Type def start? Int burst time; Int winder thre; Int queue: Int remaining time;	# time t = to: processes [1] - remaining - to: processes [1] - remaining - to: # time t = processes [1] - remaining - to: # time t = processes [1] - remaining time; processes [1] - waiting time = * time - processes processes [1] - waiting time = * time - processes processes [1] - turnaround time = * time - processes [1] - turnaround time = * time - process [1] - remaining time = 0;
April processes after the street fractions a use the processes in the processes in each queue. Solut. It include < stdio & > ## define MAX processes to ## define TIME - ANANTUM a type def start { int burst time; int without thre; int queue; int green; int green; int remaining, time;	Selse of ternal ning - by; # time += processor [i] remains time; processor [i] withing time = * time = processor processor [i] turnaround time = * time - processor [i] turnaround time = * time - frocess [i] remaining time = 0;
Franciscos in each queue. Solu? A include (stdio h) # define MAX INCECSES 10 # define TIME - QUANTUM a type def start { int id; int during time; int queue. Int remaining time; }	Selse of ternal ning - by; # time += processor [i] remains time; processor [i] withing time = * time = processor processor [i] turnaround time = * time - processor [i] turnaround time = * time - frocess [i] remaining time = 0;
Franciscos in each queue. Solu? A include (stdio h) # define MAX INCECSES 10 # define TIME - QUANTUM a type def start { int id; int during time; int queue. Int remaining time; }	processor [i] remains time; processor [i] waiting time = + time - processor burt time - processor avisation processor [i] turnaramed time = + time processor [i] avisation processor [i] avisation; processor [i] avisation;
# Include < stdio. 6> # define MAX trocesses 10 # define TIME - QUANTUM a type def start { int id; int bust time; int queue; int queue; int remaining time;	processor [i] remains time; processor [i] waiting time = + time - processor burt time - processor avisation processor [i] turnaramed time = + time processor [i] avisation processor [i] avisation; processor [i] avisation;
# Include < stdio. 6> # define MAX trocesses 10 # define TIME - QUANTUM a type def start { int id; int bust time; int queue; int queue; int remaining time;	processed the processed animalian processed animalian;
# define MAX mocesses 10 # define TIME - QUANTUM a type def struct { int burst time; int dwinal time; int queue; int remaining time;	processed autological autological processed autological autological autological autological fraction of fraction (1) autological fraction of fraction (1) autological fraction of fraction
the define TIME - QUANTUM a three def strict ? int id : int burst - time; int queue; int queue; int remaining time;	fracen (1) remaining time = 0;
type def struct ? int id ; int burst time; int queue; int queue; int memaining time;	from (1) - remaining the = 0;
typedef struct ? int barst time; int author thre; int queue; int remaining time;	}
int dural time; int guere; int gree; int gree; int gree; int gree;) the second sec
int dural time; int guere; int gree; int gree; int gree; int gree;	
int author time; int gueire; int memaining time;	The state of the s
int greve:	The second
1nt remaining time;	ste (I dane);
John out of the state of	are (, card)
int waiting time;	The state of the s
V O · · ·	I fot I know any on FT bot n bot
int turnarand time? Voi	d fefs (fraces processes [], int n, int + thre) &
in Hesponse time;	too (mt i=0; i(n; i+t) ?
Shows;	total waiting + = proconati) walting time;
0 10 10 0 50 01	
void rand stin (hocoss hocossess [7, int),	total responses + = processed total responses
mt to, Int time?	total sespon cost
int done, i;	3 Au Will - Frage " 12.25 total
do E	rand (" Average Weiting Fine's ". 2 fortal walry)
done = 1;	10 (10) Turn Asound Time 1/2 . of " total

		C toty
	Print ("Aug. Regionse hore : " = of" total reports of in Through put : " . of " stotal torseasons"	Tuonguano
	and the constraint of the state	for (Int &= 0; E(n: Ptt 1).
	the through the service and the about	for (Int &= 0; P(n = P+t);
		Procon [1] Id opporting
7/	21 1 11 5	Process [1] it process is welling the
	ant main () E	
	int main () E int n, time = 0;	groven (1) segance Him)?
-	procon proconos [MAX fracence];	
	1110	A A Street American American
	point ("Enter no of processes") record (""od", en);	(alculate average (processes, n);
The state of	searl ("Yod", en);	setus n. o.
	The state of the s	return a processor, n);
	- from (" Queux to system process in	
	I sold (" Queur to system proces in Dueve a is user proces").	output.
	The second selection of the second of the se	output.
	for (int i=0; i(n; 1++) }	
	asing the law and The and The	Enter number of processes 6.4
	print (Venter Brust Time, sorrid Time	Cheese 1 is sychem process
	early (" "d ", d ", & proconessi).	Oueve 2. 15 User Proces
	scar (rd rd god, & proconosti).	
	burst-time, & processes Pi) arrival the	Enter 87, AT and Queir of P1: 201
	& proconesti), queud;	Easter BT, AT and great of to \$1 02
	process of (1) . remains time = process time burst time	C has PF A7 and Queue of F3 & SOI
tool 1	byott Ha	Enter BT, AT and brund Py 5 302
1-10/2	(nocena (T) . waiting there = 0;	
	processo (i) ternorgind time = 0;	PROCESS WIT TAT RT
	lastones (1) a page out at a a a	1 0 2
	fascenso (i) a response time = 03	2 2 4
		7 2
-	2 12 1	3 2 1 8
	roundischin (processes, n. Time Dieter	4 1 1 0 1
	etine);	Aug. W.T. = 4.25
	fef 3 (procoses, n. 8 tine);	Avg TAT 57:00
	for s (wascosses, m, & time); frintf (\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \	Ang RT: 4025
		Thoughput = 0.57.
		(Nao agri)

Program-4

Question:

```
Write a C program to simulate Real-Time CPU Scheduling algorithms: a)Rate- Monotonic b)Earliest-deadline First c)Proportional scheduling
```

```
=>RATE-MONOTONIC

#include <stdio.h>

#define MAX_PROCESSES 10

typedef struct {
  int id;
  int burst_time;
```

```
int period;
  int remaining time;
  int next deadline;
} Process;
void sort by period(Process processes[], int n) {
  for (int i = 0; i < n - 1; i++) {
     for (int j = 0; j < n - i - 1; j++) {
        if (processes[j].period > processes[j + 1].period) {
          Process temp = processes[j];
          processes[j] = processes[j + 1];
          processes[j + 1] = temp;
     }
  }
int gcd(int a, int b) {
  return b == 0? a : gcd(b, a % b);
}
int lcm(int a, int b) {
  return (a * b) / gcd(a, b);
}
int calculate lcm(Process processes[], int n) {
  int result = processes[0].period;
  for (int i = 1; i < n; i++) {
     result = lcm(result, processes[i].period);
```

```
}
  return result;
}
double utilization factor(Process processes[], int n) {
  double sum = 0;
  for (int i = 0; i < n; i++) {
     sum += (double)processes[i].burst time / processes[i].period;
  }
  return sum;
double rms threshold(int n) {
  return n * (pow(2.0, 1.0 / n) - 1);
}
void rate monotonic scheduling(Process processes[], int n) {
  int lcm period = calculate lcm(processes, n);
  printf("LCM=%d\n\n", lcm period);
  printf("Rate Monotone Scheduling:\n");
  printf("PID Burst Period\n");
  for (int i = 0; i < n; i++) {
     printf("%d %d
                         %d\n", processes[i].id, processes[i].burst time,
processes[i].period);
  }
  double utilization = utilization factor(processes, n);
  double threshold = rms threshold(n);
```

```
printf("\n\%.6f \le \%.6f = \%.6f
? "true" : "false");
           if (utilization > threshold) {
                    printf("\nSystem may not be schedulable!\n");
                     return;
           }
          int timeline = 0, executed = 0;
           while (timeline < lcm period) {
                     int selected = -1;
                    for (int i = 0; i < n; i++) {
                                if (timeline % processes[i].period == 0) {
                                           processes[i].remaining time = processes[i].burst time;
                                 }
                                if (processes[i].remaining time > 0) {
                                           selected = i;
                                           break;
                                 }
                      }
                    if (selected != -1) {
                                printf("Time %d: Process %d is running\n", timeline, processes[selected].id);
                                processes[selected].remaining time--;
                                executed++;
                      } else {
                                printf("Time %d: CPU is idle\n", timeline);
                      }
                     timeline++;
           }
```

```
}
int main() {
  int n;
  Process processes[MAX_PROCESSES];
  printf("Enter the number of processes: ");
  scanf("%d", &n);
  printf("Enter the CPU burst times:\n");
  for (int i = 0; i < n; i++) {
     processes[i].id = i + 1;
     scanf("%d", &processes[i].burst_time);
     processes[i].remaining time = processes[i].burst time;
  }
  printf("Enter the time periods:\n");
  for (int i = 0; i < n; i++) {
     scanf("%d", &processes[i].period);
  }
  sort_by_period(processes, n);
  rate monotonic scheduling(processes, n);
  return 0;
```

```
Enter the number of processes: 3

Enter the CPU burst times:
3
6 8

Enter the time periods:
3 4 5

LCM=60

Rate Monotone Scheduling:
PID Burst Period
1 3 3
2 6 4
3 8 5

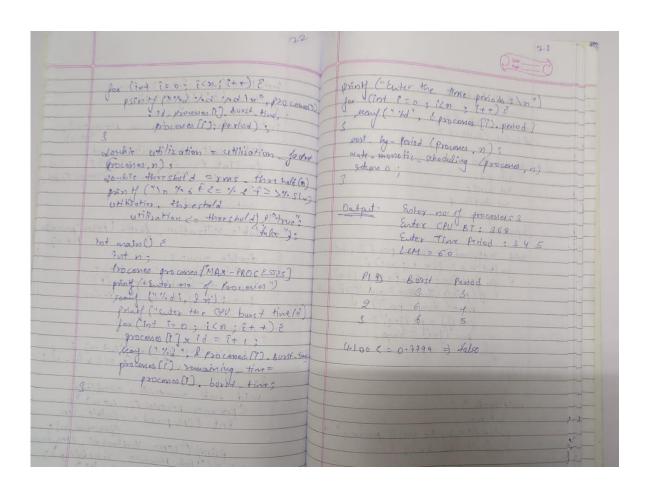
4.100000 <= 0.779763 => false

System may not be schedulable!

Process returned 0 (0x0) execution time : 18.410 s

Press any key to continue.
```

1-0-2	() box
[Lab-3.	K par = C
Rate Monotonia	just LCM (Int a
Kare isomotonia.	uetural 0 * h) /
#include (stdip. A >	First ICM (int a fint b) { Rechisen(a*b)/gcd(a,b); Sent sole B.
# MOCOGE STORE NO.	Ent calculate 1.
the define MAX Processes 10	(froup frocested)
tio tol Chart 5	3 Sont calculate tem (from processes) Int result - processes (a) x period for (int i=1: ien itt): Herult + LCM (excult, processes i) per exeture result:
typedel Struct &	for (int is) interestoly period.
int buost time;	nesult = 10m (= 1+1) &
int period;	3. (HOW) + processof i) per
Int remaining time;	Hetum verult;
Int mext deadline ;	The state of the s
3 fracon	double utilization factor (from processed
August 1	daces (1 some browning
you'd sort by period (Procen processes 13 july)	double sum = 0;
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	for lint is now a say
la (1+1=02) (2) (1+2) (1+1) 5	for lint (20 ; or n; itit) ; (um t = (louble) processes (1)
il (expense) period a constant	buset time processes [i]. period;
for (int j = 0) i (n-i) (tt) } for (int j = 0) j (n-i) (2 j+t) { (processes [j]) period > processing processing period)	(sales - max process [1] period ;
	retian sun;
process tenge = processes (p), processes (f) = processes (9+1), processes (f+D = tenge;	7
process () - process ()	double mo threshold (int m) &
5. Frommer you	double to the form of the state
(3 suction n (gow (20, 10/n-1)
,	
The state of the s	void sale monotonie scheduling
	Cprocesses processes (7, int n) &
	Ent CCM period = calculate ich
Int gcd (int 9, int b) &	(processor 2);
Int gcd (int 9, int b) 5 getum b = = 0 ? a : gcd (b, a 1.6);	grant ("ICM= "1. d) n / CM- pa
3	
10 F 2 T 4T 34	Print (" PID Borst Period 19th)
A X 72 VA	



=>EARLIEST-DEADLINE

```
#include <stdio.h>
#include <stdib.h>

typedef struct {
   int pid;
   int burst;
   int deadline;
   int period;
} Process;

void sortByDeadline(Process proc[], int n) {
```

```
for (int i = 0; i < n - 1; i++) {
     for (int j = 0; j < n - i - 1; j++) {
        if (proc[j].deadline > proc[j + 1].deadline) {
          Process temp = proc[j];
          proc[j] = proc[j + 1];
          proc[j + 1] = temp;
     }
  }
void main() {
  int n;
  printf("Enter the number of processes:");
  scanf("%d", &n);
  Process proc[n];
  printf("\nEnter the CPU burst times:\n");
  for (int i = 0; i < n; i++) {
     scanf("%d", &proc[i].burst);
     proc[i].pid = i + 1;
  }
  printf("\nEnter the deadlines:\n");
  for (int i = 0; i < n; i++) {
     scanf("%d", &proc[i].deadline);
  }
  printf("\nEnter the time periods:\n");
  for (int i = 0; i < n; i++) {
     scanf("%d", &proc[i].period);
  }
```

```
sortByDeadline(proc, n);
printf("\nEarliest Deadline Scheduling:\n");
printf("PID Burst Deadline Period\n");
for (int i = 0; i < n; i++) {
    printf("%d %d %d %d\n", proc[i].pid, proc[i].burst,
proc[i].deadline, proc[i].period);
}
printf("\nScheduling occurs for 6 ms\n");
for (int time = 0; time < 6; time++) {
    printf("%dms : Task %d is running.\n", time, proc[0].pid);
}</pre>
```

```
Enter the number of processes:3
Enter the CPU burst times:
2 3 5
Enter the deadlines:
1 2 3
Enter the time periods:
1 2 3
Earliest Deadline Scheduling:
       Burst Deadline
                              Period
       2
                 1
       3
                 2
                             2
                 3
Scheduling occurs for 6 ms
Oms : Task 1 is running.
1ms : Task 1 is running.
2ms : Task 1 is running.
3ms : Task 1 is running.
4ms : Task 1 is running.
5ms: Task 1 is running.
Process returned 6 (0x6)
                             execution time : 17.130 s
Press any key to continue.
Process returned 0 (0x0)
                             execution time : 28.063 s
Press any key to continue.
```

for (int i= 0; ic n i it r) i heart ("" led", I process hours): process plat it i; Earliest deadline # melude < stdio . h > typedel struct & frinty ("Enter the time deadlines sin"); for (int 1= 0; ((n : 1++) ; Read ("Ved", a prot (1). deadline); and pld; int burst; Int deadline; Int period ; fort (" ever the soutine periods = \n"); for (Int i= 0; i(n; i+ 1); scarf (" glad", l prollipoba); void contentrabline (process proc [], int n) &

for (int to 0; i(m-1; i+t) &

for (int f=0; i(m-i-t); j+t) &

if (proc []) deadline > proc (ftill adeadlin)

proc []) = proc [];

proc []+ [] = kup; Sort By Deadline (proc , n); print ("Earlist Deadline Scheduling (12")

print ("PID Burst Deadline leved In')

for (int i=0; "in ; 7++) i

frint (""Lad "id "id "in procliped,

proc (I), burst, proclipedeadlise,

proc (I), burst, proclipedeadlise, yord main O ? Int n;
frint ("Enter the no. of process");
Keary ("led", In); frint ("In Scheduling occurs for ans In"); 1.
for (int tincso); time (8; times)
print ("bd ms: Task "Ind is aumning la";
the proclayed proces proc(n); printy (Enter the CPU board times : \all);

Enter the no. of grocenes = 3

Enter the clos borst Tines = 2

Enter the deadlines = 1 2

Enter the time = 1 2

Enter the time = 1 2

Earliest Deadline Scheduling

PID Rurst Peadline Period

1 2 3

3 2 2

3 3 2 3

Staduling recess for 6 ms

ens = Task | 1 s runing

Ins = 5 m;

ym;

fracessos exactision time = 12.120 =

fraces = channed crectution time = 28.0000

Question:

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

=>PRODUCER-CONSUMER

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
int mutex = 1, full = 0, empty = 3, x = 0, buffer = 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 = rand() \% 50;
  buffer = x;
  printf("Producer 1 produced %d\n", x);
  printf("Buffer:%d\n", buffer);
```

```
mutex = signal(mutex);
}
void consumer() {
  mutex = wait(mutex);
  full = wait(full);
  empty = signal(empty);
  printf("Consumer 2 consumed %d\n", buffer);
  printf("Current buffer len: 0\n");
  X--;
  mutex = signal(mutex);
}
void main() {
  int choice, p, c;
  srand(time(0));
  printf("Enter the number of Producers:");
  scanf("%d", &p);
  printf("Enter the number of Consumers:");
  scanf("%d", &c);
  printf("Enter buffer capacity:");
  scanf("%d", &empty);
  for (int i = 1; i \le p; i++)
     printf("Successfully created producer %d\n", i);
  for (int i = 1; i \le c; i++)
     printf("Successfully created consumer %d\n", i);
  while (1) {
    printf("\n1.Producer\n2.Consumer\n3.Exit\n");
```

```
scanf("%d", &choice);
switch (choice) {
  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);
}
```

```
Enter the number of Producers:1
Enter the number of Consumers:1
Enter buffer capacity:1
Successfully created producer 1
Successfully created consumer 1
1.Producer
2.Consumer
3.Exit
Producer 1 produced 17
Buffer:17
1.Producer
2.Consumer
3.Exit
2
Consumer 2 consumed 17
Current buffer len: 0
1.Producer
2.Consumer
3.Exit
Producer 1 produced 16
Buffer:16
1.Producer
2.Consumer
3.Exit
Consumer 2 consumed 16
Current buffer len: 0
1.Producer
2.Consumer
3.Exit
```

Roducer Consumes.

Include (station is to include (station is to include (station is to time the station is to time the station is to the station of the s

		output ?
Fort main() &		
Sud pholog s		(H) Producer
1 5		(2) Consumer
nt 1 m produces in	Consumer)	(2) cuit
end s		Enter your choice 1
4		produce of produces that items!
a ent (b suter choice)		
st Ladog & g ch);		Exter Ch 1
switch (choice) ?		produces produces items
Care (1)		The same of the same
if muten == 1) & en	12+4	Enter ch 1
produce ()		
1 12 or th I nowhard		producer produces item 3
else ; (Boffer is full");		Entes your clusice is !
break;		Buffer is full
poetas)		
		Entor your choice 52
Case & C If (mate x = 1) ll (full	= 1 3	Consures Consures items 3
17 (mote n - 1) 24 (701)	-0/	(04344
consume ()?		-0.1
c fs e		Enter your choice a
pf (Boffer is en	pty")	Enter your relation 22
break;		
Cuse 3 %		
exit (0);		
default &		
pf ("Invalid Choice"	112	
netunn o;		
3		

Question:

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

```
=>DINING-PHILOSOPHER
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
#include <semaphore.h>
#include <unistd.h>
int n;
sem t*chopstick;
pthread t *philosopher;
int *phil ids;
void* philosopher fn(void* num) {
  int i = *(int*)num;
  int left = i;
  int right = (i + 1) \% n;
  printf("Philosopher %d is hungry\n", i + 1);
  sem wait(&chopstick[left]);
  sem wait(&chopstick[right]);
  printf("Philosopher %d took chopstick %d and %d\n", i + 1, left + 1, right + 1);
  printf("Philosopher %d is eating\n", i + 1);
  sleep(1);
```

```
printf("Philosopher %d finished eating and kept the chopstick back on table\n", i
+1);
  sem post(&chopstick[left]);
  sem post(&chopstick[right]);
  return NULL;
}
int main() {
  int hungry, i;
  printf("Enter the number of philosophers:");
  scanf("%d", &n);
  chopstick = malloc(n * sizeof(sem_t));
  philosopher = malloc(n * sizeof(pthread t));
  phil ids = malloc(n * sizeof(int));
  for (i = 0; i < n; i++) sem init(&chopstick[i], 0, 1);
  printf("Enter number of hungry philosophers:");
  scanf("%d", &hungry);
  for (i = 0; i < \text{hungry}; i++)  {
     phil ids[i] = i;
     pthread create(&philosopher[i], NULL, philosopher fn, &phil ids[i]);
     sleep(1);
  }
  for (i = 0; i < \text{hungry}; i++) 
     pthread join(philosopher[i], NULL);
  }
  free(chopstick);
  free(philosopher);
  free(phil ids);
  return 0;
```

}

```
Enter the number of philosophers:5
Enter number of hungry philosophers:5
Philosopher 1 is hungry
Philosopher 1 took chopstick 1 and 2
Philosopher 1 is eating
Philosopher 1 finished eating and kept the chopstick back on table
Philosopher 2 is hungry
Philosopher 2 took chopstick 2 and 3
Philosopher 2 is eating
Philosopher 2 finished eating and kept the chopstick back on table
Philosopher 3 is hungry
Philosopher 3 took chopstick 3 and 4
Philosopher 3 is eating
Philosopher 3 finished eating and kept the chopstick back on table
Philosopher 4 is hungry
Philosopher 4 took chopstick 4 and 5
Philosopher 4 is eating
Philosopher 4 finished eating and kept the chopstick back on table
Philosopher 5 is hungry
Philosopher 5 took chopstick 5 and 1
Philosopher 5 is eating
Philosopher 5 finished eating and kept the chopstick back on table
Process returned 0 (0x0)
                            execution time: 8.406 s
Press any key to continue.
```

P bots of (! are neighbor [lungry ti);

pf ("Comb. %d 121", Combination

pf ("Y-d & p 7 d are granted to cat 1"

hugry [i), Engry [j] Dinning Philosopher # include < stain h> # Include < st dlib. h > # define man 10: for (int K=0; K(cont; K+1) E int hungy FMAX];

Int and Neighbours (Inta, Intb) E

sectors (ab \$(a-b) = = 1 (1 abs (a-b)=a

total phil 1); int total Philosopher ? print (pokd is mad writingly huggette) print ("1" ") , 33] void option (int count) ? pt ("flow one philosopher to eat

at any time ln');

for (int i = 0; iccount; (++);

fring ["id is granted to eat").

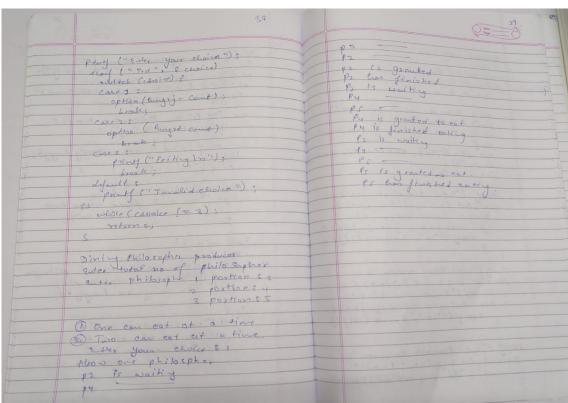
for (int je o; jccount; fr+);

for (int je o; jccount; fr+);

for (int je o; jccount; fr+); of (combination == 1) ?

of (no comb found where two non-neighbors cat him); 20 printf ("Exter the total mo. of philosopher read (""d", I totalphilosopher (""Now many are hungry"); print (" P ded is wasting 4) 3133 huggy(j)); void options (Int count) & pf (" one can gat at a time) print (" in a low two philospher peat at some time In") 2) two can car at a time for (int f = (+ () < count)

for (int f = (+ () < count) 2) cexit



Question:

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

```
=>BANKERS ALGORITHM
#include <stdio.h>
#include <stdbool.h>
void main() {
  int n, m;
  printf("Enter number of processes and resources:\n");
  scanf("%d %d", &n, &m);
  int alloc[n][m], max[n][m], avail[m];
  printf("Enter allocation matrix:\n");
  for (int i = 0; i < n; i++)
     for (int j = 0; j < m; j++)
       scanf("%d", &alloc[i][j]);
  printf("Enter max matrix:\n");
  for (int i = 0; i < n; i++)
     for (int j = 0; j < m; j++)
       scanf("%d", &max[i][j]);
  printf("Enter available matrix:\n");
  for (int i = 0; i < m; i++)
     scanf("%d", &avail[i]);
```

```
int finish[n];
int safeSeq[n];
for (int i = 0; i < n; i++) finish[i] = 0;
int need[n][m];
for (int i = 0; i < n; i++)
  for (int j = 0; j < m; j++)
     need[i][j] = max[i][j] - alloc[i][j];
int count = 0;
while (count \leq n) {
  bool found = false;
  for (int i = 0; i < n; i++) {
     if (!finish[i]) {
        int j;
        for (j = 0; j < m; j++)
          if (need[i][j] > avail[j])
             break;
        if (j == m) {
          for (int k = 0; k < m; k++)
             avail[k] += alloc[i][k];
          safeSeq[count++] = i;
           finish[i] = 1;
           found = true;
  if (!found) break;
}
if (count == n) {
```

```
printf("System is in safe state.\n");
printf("Safe sequence is: ");
for (int i = 0; i < n; i++) {
    printf("P%d", safeSeq[i]);
    if (i != n - 1) printf(" -> ");
}
printf("\n");
} else {
    printf("System is not in a safe state.\n");
}
```

```
C:\Users\Admin\Desktop\ban X
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 max matrix:
7 5 3
3 2 2
9 0 2
2 2 2
4 3 3
Enter available matrix:
3 3 2
System is in safe state.
Safe sequence is: P1 -> P3 -> P4 -> P0 -> P2
Process returned 10 (0xA)
                              execution time : 159.943 s
Press any key to continue.
```

Banker's Dig a.

It include (station is to

It include (station is to

Int no m

printf ("enter no. of processes & sessi)

enof ("led "od", & n, 2m);

int allow [n] [m], answ [n] [m], avail(s)

printf ("enter allowation and in");

for (Int j = 0; l(n; i+t)

for (Int j = 0; l(n; i+t)

for (Int j = 0; l(n; i+t)

coay ("", d", l avail (i));

int speed (n);

for (int i = 0; l(n; i+t)

for (int i = 0; l(n; i+t)

for (int j = 0;

	29 109
	Owtput:
bool lound = Palse;	Com
los (int i=0 ; iln; i+1) &	a day
pool bound = false; food (int iso s film; itt) { if (!finish (i)) { int j;	Enter number of passacres and resources:
int ? ;	to tax and many
int j: for (se a s sem if + t) (need (r) (j) \ availty) break;	Suter allocation making
1) (need (F) (9) savailte)	200
break;	302
il (s== m) ?	211
for (int K=0; K(m; K+t)	
grail[k] += alloe [i][x]-	0 0 2
lafered [count + +) =1;	Enter max matrix:
Cinick [i] = 1	7 3 3
found = true;	3 2 2
- In a second to the second to	902
- 1 CT CT CON SOME STORY OF THE	2 2 2
2	4 3 3
if (I found) break;	Enter Available Matrix:
2 (Journ) billion,	3 3 2 morna !
Comment of the second	Sulter & S. A.
21 / 11 7 5	Sylten is in safe state.
if (court = = n) {	safe source sequence is finf the sport
print (System is in safe state in);	0 1: 11
grinf (Safe Sequence is : ");	Execution time 2 61.800 5
po (int = 0 3 i(n; i+t)]	the first to a fector since
grint 1 "pelad", lede seg[7]):	
31 (4/= n-1) month (" -> ")	
print ("System is in safe glate"): print ("Safe segunce is "); pro (int i= 0 3 i(n ; i++) ; print ("Pond", life seg[1]); if (1/= n-1) print ("");	
3 clse 2 mit ("\n");	and the same of th
	and the season of tall of
S USE ? IT SOME THE PROPERTY OF THE PROPERTY O	
print ("System is not in safe state");	
3	

```
Program-8
Question:
Write a C program to simulate deadlock detection
=>DEADLOCK
#include <stdio.h>
#include <stdbool.h>
int main() {
  int n, m;
  printf("Enter number of processes and resources:\n");
  scanf("%d %d", &n, &m);
  int alloc[n][m], request[n][m], avail[m];
  printf("Enter allocation matrix:\n");
  for (int i = 0; i < n; i++)
     for (int j = 0; j < m; j++)
       scanf("%d", &alloc[i][j]);
  printf("Enter request matrix:\n");
  for (int i = 0; i < n; i++)
     for (int j = 0; j < m; j++)
       scanf("%d", &request[i][j]);
  printf("Enter available matrix:\n");
  for (int i = 0; i < m; i++)
     scanf("%d", &avail[i]);
  int finish[n];
  for (int i = 0; i < n; i++) finish[i] = 0;
  int count = 0;
  while (count \leq n) {
```

```
bool found = false;
  for (int i = 0; i < n; i++) {
     if (!finish[i]) {
        int j;
        for (j = 0; j < m; j++)
          if (request[i][j] > avail[j])
             break;
        if (j == m) {
          for (int k = 0; k < m; k++)
             avail[k] += alloc[i][k];
          finish[i] = 1;
          printf("Process %d can finish.\n", i);
          count++;
          found = true;
     }
  if (!found) break;
}
if (count == n)
  printf("System is not in a deadlock state.\n");
else
  printf("System is in a deadlock state.\n");
return 0;
```

```
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 0 2
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: 75.157 s
Press any key to continue.
```

		La G
		farlint is a silen site of sint just just just just just just just jus
	Deadlock Avoidance	(1 (10) 1 (n s (++) 5
		int int (i)) E
	# include < stdlo . h >	Jac(50) J(m; 1+1) If (request() (f) \ avii(f) break; If (= m) E
-	# Include < std 600l. R'Y	3/ (m. 9+4)
	3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	J (request(i) Frence at 2)
	Int main () E	break; Jas Mail (j.)
	int n, m;	if (i m) E
	prints ("Enter no of processes & resources)	to the kind of the
	real (14d 1/2d , en, 8m) 3	quailfk) + = allactions;
		finish (T) = 2;
	int allow [n] [m], request [n][m], availlin]	print (" process of 100 cm
	Print (Enter alleation matrix);	print (" Procenced complinist)
	for (int i=0; i(n; 1++)	e found = true .
	for (int 1=0 : 1 (m 5 1++)	30
	for (int j=0; j (m 5 j++) (m 5 j);	
	can con con	1.
	and al / " Enter room of water classes	91 (11, 1) 4.4
	print (" Enter request matrix: \n");	3 y (found) break;
	for (int 1=03 i < n s j+t)	
	for (int j = 0; (m;)++)	y (count == n)
3	for (" 1 = 0;) < m;)++); leaf (" 10d", & request [1][]);	prints ("gystem is not in a dealback")
	print ("Enter ovailable matrix"); for (int 6 0 3 Rem 3 (+1) leaf ("Id", & avail (1)));	part ("system is in deadlack state")
1	low (int 10 0 3 Pem 5 E+r)	section o;
	1 ("1/2" 9 - 31 557 1) 3	AND DE COMMENTAL D
	lead a salation (12 22)	1
	1 10 0 1 5 0	
- 1 5	int finish [n);	
	(or (int i=0; 7(n; i++) finish pi)=+	
Pu	of count = 0;	
- u	shile (court (n) i	
	bool found = false;	

```
Enter number of processes 2 resource

5 3

Enter Allocation malrix:

0 20 2

2 L IV

0 0 0 2

2 L IV

1 2 L IV

1 2 2 2

2 2 2 2

3 0 2

2 2 2 2

4 2 3

Enter avoilable motive:

3 3 2

frocon 1 cuan flowish

- 2 conf flowish

- 4 cans flowish

system 1s in a deadlack state

Cx cellton thre 2 75.1575
```

Question:

```
Write a C program to simulate the following contiguous memory allocation techniques a)
Worst-fit
-Best-fit
-First-fit
=>MEMORY ALLOCATION
```

```
struct Block {
  int size;
  int allocated;
};
```

#include <stdio.h>

```
struct File {
  int size;
  int block no;
};
void resetBlocks(struct Block blocks[], int n) {
  for (int i = 0; i < n; i++) {
    blocks[i].allocated = 0;
  }
}
void firstFit(struct Block blocks[], int n blocks, struct File files[], int n files) {
  printf("\n\tMemory Management Scheme – First Fit\n");
  printf("File no:\tFile size\tBlock no:\tBlock size:\n");
  for (int i = 0; i < n files; i++) {
    files[i].block no = -1;
    for (int j = 0; j < n blocks; j++) {
       if (!blocks[j].allocated && blocks[j].size >= files[i].size) {
         files[i].block no = j + 1;
         blocks[i].allocated = 1;
         break;
    if (files[i].block no == -1) {
       printf("%d\t\t%d\t\t \\t\ \n", i + 1, files[i].size);
    }
  }
```

```
}
void bestFit(struct Block blocks[], int n blocks, struct File files[], int n files) {
  printf("\n\tMemory Management Scheme – Best Fit\n");
  printf("File no:\tFile size\tBlock no:\tBlock size:\n");
  for (int i = 0; i < n files; i++) {
     int bestIdx = -1;
     for (int j = 0; j < n blocks; j++) {
       if (!blocks[j].allocated && blocks[j].size >= files[i].size) {
          if (bestIdx == -1 || blocks[j].size < blocks[bestIdx].size) {
             bestIdx = i;
       }
     if (bestIdx != -1) {
       blocks[bestIdx].allocated = 1;
       files[i].block no = bestIdx + 1;
       printf("%d\t\d\t\t%d\t\t%d\n", i + 1, files[i].size, bestIdx + 1,
blocks[bestIdx].size);
     } else {
       printf("%d\t\t%d\t\t \h", i + 1, files[i].size);
     }
}
void worstFit(struct Block blocks[], int n blocks, struct File files[], int n files) {
  printf("\n\tMemory Management Scheme – Worst Fit\n");
  printf("File no:\tFile size\tBlock no:\tBlock size:\n");
```

```
for (int i = 0; i < n files; i++) {
     int worstIdx = -1;
     for (int j = 0; j < n blocks; j++) {
       if (!blocks[j].allocated && blocks[j].size >= files[i].size) {
          if (worstIdx == -1 || blocks[j].size > blocks[worstIdx].size) {
            worstIdx = j;
          }
       }
     if (worstIdx !=-1) {
       blocks[worstIdx].allocated = 1;
       files[i].block no = worstIdx + 1;
       printf("%d\t\d\d\", i + 1, files[i].size, worstIdx + 1,
blocks[worstIdx].size);
     } else {
       printf("%d\t\t%d\t\t \n", i + 1, files[i].size);
}
int main() {
  int n blocks, n files, choice;
  printf("Memory Management Scheme\n");
  printf("Enter the number of blocks: ");
  scanf("%d", &n blocks);
  printf("Enter the number of files: ");
  scanf("%d", &n files);
  struct Block blocks[n blocks];
  struct File files[n files];
```

```
printf("\nEnter the size of the blocks:\n");
for (int i = 0; i < n blocks; i++) {
  printf("Block %d: ", i + 1);
  scanf("%d", &blocks[i].size);
  blocks[i].allocated = 0;
}
printf("Enter the size of the files:\n");
for (int i = 0; i < n files; i++) {
  printf("File %d: ", i + 1);
  scanf("%d", &files[i].size);
}
do {
  printf("\n1. First Fit\n2. Best Fit\n3. Worst Fit\n4. Exit\n");
  printf("Enter your choice: ");
  scanf("%d", &choice);
  resetBlocks(blocks, n blocks); // Reset block allocation before each strategy
  switch (choice) {
     case 1:
       firstFit(blocks, n blocks, files, n files);
       break;
     case 2:
       bestFit(blocks, n blocks, files, n files);
       break;
     case 3:
       worstFit(blocks, n blocks, files, n files);
       break;
     case 4:
       printf("\nExiting...\n");
```

```
break;
default:
    printf("Invalid choice.\n");
}
} while (choice != 4);
return 0;
}
```

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

    First Fit

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

    First Fit

2. Best Fit
3. Worst Fit
4. Exit
Enter your choice: 🕳
```

Labora.

Lab

(best Idm 1 = 1) E blocks [best Jan] allocated = 1: Fres (1) back no - j+1;

Backs (1) allocated = 1;

pantf (1) d t + 1 ind t + 1 ind it + 2 ind it + 2);

pantf (1) d t + 1 ind t + 1 ind it + 2 ind it + 2);

placks (1) size, j+1;

placks (1) size; blocks I best Is my allocated

files (i). blocks no best Idn +1;

fries (i) give, best Idn + 1, blocks

(best Idn) sine break; [| files [i] . block - no == -1) [

printf ("d | t | t | d | t | t | t | no)

ft | feles [i] , size); void worst Fit (struct Block blocks [] int n blocks, struct file Files). in+ n-tiles) [. frint (") mH Memory Hangewest scheme" void best fit (struct Block blocks 1) Int nublocks, struct files files () Ent n files) & print ("In) + Memon Management schen for (Int i= 0; i/n liles; it+) & printf (" file no, It file size It Block size of for lint j= 0 jem black, j++) & blocks [j], size >= files[j]. for (int i= 0; i < m. files; i++) 5 mit best Idn -- 1; blocks [worst Idn). allocated - 1 for (int j= 0; jcn-blocks; j+t)! y (! Hocks [j). allocated 80 blocks [j) size >= #los[i). files [1] block - no = worst Janti prints ("1.d) t \t v.d \t v.d \s) y Chest Idn == 1) ; best Idx = j. blocks [worst Idn) six); print (" " d H H 2 d H H W

	The state of the s
	P box
not maint) &	Output.
int on blacks in tiles, choice	S. day
pro 19 ("Memory Management Scheme")	Enter the no of blocks: 4 Enter the no of blocks: 4
print ("Enter the no of blocks")	The parafiles 35
scarf (47.d", & n blocks);	Souther the 2
printy (" suster the size of files")	Bis 20 She of the blocky:
for (int) i= 0; i(n plus; f++); print ('file ad', ++);	B 2 ° 15
grand ("Ed , 2 filos [i) size)	B 3 : 30
1 (and (a 1 1 1 as (1 2 as 1 2 a)	B 4:5
2 + broke and an	and the state of t
do 8	
printf ("In I first In 2. Best Fith	Enter the size of the files:
White was the track of	£ 2 5 5
Printle (" Entre your choice")	f 2 , 20
econy (4 % d & Schoice)	f y · 25
excet Blocks (blocks, n. block).	£ 5: 40
Switch (choice) ?	
Case 1:	f 17 fixit fit
15-Hit / blocks at the let 10	P) Best fit
firstit (blocks, m-blocks, files, n/is)	(2) worst fit
P. C.	
Case 2:	(v) Enit.
bestfit (blacks, n. blacks, file, n. file	The state of the s
& break ;	Enter choice: 1
Case 2:	Tank at 11 th 12 cape of
worst fit (blocks, nblocks, file, 263)	file no. The size Block no. Block 820
bocaks	1 10 1 20
Case 4:	2 5 2 15
print ("In Exiting -o- \n");	3 30 30 50
bree k:	y 25
	5 40
defaults (" Invalid choice")	
	Enter choices 4
3 while (choice (=4)	Exiting

```
Question:
Write a C program to simulate page replacement algorithms a) FIFO -LRU -Optimal

=>LRU
#include <stdio.h>

int main() {
    int n, frames, i, j, k, faults = 0;
    printf("Enter number of pages: ");
    scanf("%d", &n);
    int pages[n];
    printf("Enter the reference string: ");
```

```
for(i = 0; i < n; i++)
  scanf("%d", &pages[i]);
printf("Enter number of frames: ");
scanf("%d", &frames);
int frame arr[frames];
int time[frames];
for(i = 0; i < frames; i++)  {
  frame arr[i] = -1;
  time[i] = 0;
}
int counter = 0;
for(i = 0; i < n; i++) {
  int flag = 0;
  for(j = 0; j < \text{frames}; j++) {
     if(frame_arr[j] == pages[i]) {
       flag = 1;
       counter++;
       time[j] = counter;
       break;
     }
  }
  if(flag == 0) {
     faults++;
     int min time = time[0], min pos = 0;
     for(k = 1; k < \text{frames}; k++ \}
       if(time[k] < min time) {
          min time = time[k];
          min pos = k;
```

```
}
       frame arr[min pos] = pages[i];
       counter++;
       time[min_pos] = counter;
     printf("Frames after accessing %d: ", pages[i]);
     for(j = 0; j < \text{frames}; j++) {
       if(frame\_arr[j] == -1)
          printf("- ");
       else
          printf("%d ", frame_arr[j]);
     }
     printf("\n");
  printf("Total page faults: %d\n", faults);
  int Hits = n-faults;
  printf("Total page Hits: %d\n",Hits);
  return 0;
}
```

```
Enter number of pages: 20
Enter the reference string: 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1
Enter number of frames: 4
Frames after accessing 7: 7
Frames after accessing 0: 7 0 -
Frames after accessing 1: 7 0 1 -
Frames after accessing 2: 7 0 1 2
Frames after accessing 0: 7 0 1 2
Frames after accessing 3: 3 0 1 2
Frames after accessing 0: 3 0 1 2
Frames after accessing 4: 3 0 4 2
Frames after accessing 2: 3 0 4 2
Frames after accessing 3: 3 0 4 2
Frames after accessing 0: 3 0 4 2
Frames after accessing 3: 3 0 4 2
Frames after accessing 2: 3 0 4 2
Frames after accessing 1: 3 0 1 2
Frames after accessing 2: 3 0 1 2
Frames after accessing 0: 3 0 1 2
Frames after accessing 1: 3 0 1 2
Frames after accessing 7: 7 0 1 2
Frames after accessing 0: 7 0 1 2
Frames after accessing 1: 7 0 1 2
Total page faults: 8
Total page Hits: 12
Process returned 0 (0x0)
                                execution time : 54.602 s
Press any key to continue.
```

```
=>OPTIMAL

#include <stdio.h>

int main() {
  int n, frames, i, j, k, faults = 0;
  printf("Enter number of pages: ");
  scanf("%d", &n);
  int pages[n];
```

```
printf("Enter the reference string: ");
for(i = 0; i < n; i++)
  scanf("%d", &pages[i]);
printf("Enter number of frames: ");
scanf("%d", &frames);
int frame arr[frames];
for(i = 0; i < \text{frames}; i++)
  frame arr[i] = -1;
for(i = 0; i < n; i++) {
  int flag = 0;
  for(j = 0; j < \text{frames}; j++) {
     if(frame arr[j] == pages[i]) {
        flag = 1;
        break;
     }
   }
  if(flag == 0)  {
     faults++;
     int pos = -1;
     for(j = 0; j < \text{frames}; j++) {
        if(frame_arr[j] == -1) {
          pos = j;
          break;
     if(pos == -1)  {
        int farthest = i, replace index = 0;
        for(j = 0; j < frames; j++) {
          int found = 0;
```

```
for(k = i + 1; k < n; k++) {
             if(frame\_arr[j] == pages[k]) {
               if(k > farthest) {
                  farthest = k;
                  replace_index = j;
               found = 1;
               break;
          if(!found) {
             replace_index = j;
             break;
          }
       pos = replace index;
     frame_arr[pos] = pages[i];
  }
  printf("%d: ", pages[i]);
  for(j = 0; j < frames; j++) {
     if(frame_arr[j] == -1)
       printf("_ ");
     else
       printf("%d ", frame arr[j]);
  }
  printf("\n");
printf("Total page faults: %d\n", faults);
```

```
int Hits = n-faults;
  printf("Total page Hits: %d\n",Hits);
  return 0;
}
```

```
Enter number of pages: 20
Enter the reference string: 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1
Enter number of frames: 4

7: 7
0: 7 0 - -
1: 7 0 1 -
2: 7 0 1 2
0: 7 0 1 2
0: 3 0 1 2
0: 3 0 1 2
4: 3 0 4 2
2: 3 0 4 2
3: 3 0 4 2
2: 3 0 4 2
3: 3 0 4 2
2: 3 0 4 2
1: 1 0 4 2
2: 1 0 4 2
0: 1 0 7 2
1: 1 0 7 2
Total page faults: 8
Total page Hits: 12

Process returned 0 (0x0) execution time: 29.373 s
  Process returned 0 (0x0)
Press any key to continue.
                                                                                                           execution time : 29.373 s
```

Labora frace - Degram to simulton page

(Losik a of program to simulton page (Losik a of page (Losik a of page (Losik a of page)))

(Losik a of program to simulton page (Losik a of page))

(Losik a of program to simulton page (Losik a of page))

(Losik a of program to simulton page (Losik a of page))

(Losik a of program to simulton page (Losik a of page))

(Losik a of program to simulton page (Losik a of page))

(Losik a of program to simulton page)

(Losik a of p

if (! nit) & for (int j=1; g(cop; j+c))

for (int j=1; g(cop; j+c))

for = j: int j = predict (pageoft, m, ltl, capaty , page faults ++; time ++
for (1,4) = pagenti)
exceed flow = time pant ("Oftimal Page Faults id, Page Hits I'd, Page faults, Page lits); 1 Page - faults ; Vold by (int pages f), int n, interpolity
int page joults = 0
int page aits = 0
int time = 0 Print ("LRO" Page faults : 171d,

Page Hits "id, page faults tils); int man () & for (int i= 0 ; il capacity; j++) ? int n, cal;
print ("Enter the size of pages")
SFB ("xd", ln) for (1) = -1; for (int i= 0; ((n; i+F) { bod hit = false; for (Int g = 0; j(cap ; j++)) pf ("inter the pages Arings In");
for (int i = 0; i (n ; i++);

(""/d", opage [i))

pf ("suter no of page frames.

5x (""/ad", scap) time ++; page - hits ++; bit = true; break; file (pages, n, capacity)
oftime (pages, n, cap)

Pru (pages, n, cap);

Return o;

Profoce Bl RHERI

Output

Suter the size of pages 57

Cutes the pages strings 2

Lute: the no. of page frames:

FIFO: 6 PHip: 1

Def: 5 Ph: 2

CRU: 7 PH: 0