BACHELOR OF COMPUTER APPLICATION LAB MANUAL

4th Semester

Prepared By

Pure and Applied Science Dept.

Computer Application

MIDNAPORE CITY COLLEGE

INSTRUCTIONS TO STUDENTS

- Before entering the lab, the student should carry the following things (MANDATORY)

1. Identity card issued by the college.
2. Class notes
3. Lab observation book
4. Lab Manual
5. Lab Record

- Student must sign in and sign out in the register provided when attending the lab session without fail.
- Come to the laboratory in time. Students, who are late more than 10 min., will not be allowed to attend the lab.
- Students need to maintain 80% attendance in lab if not a strict action will be taken.
- All students must follow a Dress Code while in the laboratory.
- Foods, drinks are NOT allowed.
- All bags must be left at the indicated place.
- Refer to the lab staff if you need any help in using the lab.
- Respect the laboratory and its other users.
- Workspace must be kept clean and tidy after experiment is completed.
- Read the Manual carefully before coming to the laboratory and be sure about what you are supposed to do.
- Do the experiments as per the instructions given in the manual.
- Copy all the programs to observation which are taught in class before attending the lab session.
- Students are not supposed to use floppy disks, pen drives without permission of lab- in charge.
- Lab records need to be submitted on or before the date of submission.

DATABASE MANAGEMENT SYSTEM LABORATORY MANUAL (Course Code: BCAHMJ05P)

OVERVIEW OF Oracle 10g -- Installation & study about creating, inserting and storing record CREATE TABLE

create table msc(roll numeric(2), fname varchar(20), lname varchar(15));

INSERT INTO TABLE

insert into msc values(3,'Ganesh','Santra');
insert into msc values(2,'Rajat','Santra'); insert into msc values(1,'Ashim','Sarkar'); insert into msc values(4,'Bitu','Das');

VIEW THE TABLE

select * from msc;

OUTPUT

ROLL	FNAME	LNAME
3	Ganesh	Santra
2	Rajat	Santra
1	Ashim	Sarkar
4	Bitu	Das

1. **Create Student Table (Create, Insert, view, add, update, delete and conditions) CREATE**

create table student2(name varchar(15),roll numeric(3), addr varchar(25),phn_no numeric(10),email varchar(20),dob varchar(8),age numeric(3),marks numeric(5),aadhar_no numeric(15));

INSERT

insert into student2 values('KRISHNENDU',06,'Midnapore',9064557423,'c25@gmail.com','30/04/01',21,90, 1234567890);
insert into student2 values('RAJAT',08,'Kharagpur',9876543210,'rajat@gmail.com','07/05/98',24,80, 0987654321);
insert into student2 values('GANESH',05,'Chondrokona',8798654312,'yahoo@gmail.com','09/06/99',23,70, 6789054321);
insert into student2 values('BITU',04,'123 CT Road',9087123456,'hike@gmail.com','06/03/01',21,60,9081234567); insert into student2

values('ASIM',02,'Kharagpur',6294567800,'asim@gmail.com','12/12/02',20,80,0
987123400);
insert into student2
values('ARPAN',03,'Ghatal',7865432190,'arpan@gmail.com','08/11/97',23,70,5
678094312, 'Botany');

VIEW

select * from student2;

NAME	ROLL	ADDR	PHN_NO	EMAIL	DOB	AGE	MARKS	AADHAR_NO
KRISHNENDU	6	Midnapore	9064557423	c25@gmail.com	30/04/01	21	90	1234567890
RAJAT	8	Kharagpur	9876543210	rajat@gmail.com	07/05/98	24	80	987654321
GANESH	5	Chondrokona	8798654312	yahoo@gmail.com	09/06/99	23	70	6789054321
BITU	4	123 CT Road	9087123456	hike@gmail.com	06/03/01	21	60	9081234567
ASIM	2	Kharagpur	6294567800	asim@gmail.com	12/12/02	20	80	987123400

ADD

alter table student2 add department varchar(20);

NAME	ROLL	ADDR	PHN_NO	EMAIL	DOB	AGE	MARKS	AADHAR_NO	DEPARTMENT
KRISHNENDU	6	Midnapore	9064557423	c25@gmail.com	30/04/01	21	90	1234567890	-
RAJAT	8	Kharagpur	9876543210	rajat@gmail.com	07/05/98	24	80	987654321	-
GANESH	5	Chondrokona	8798654312	yahoo@gmail.com	09/06/99	23	70	6789054321	-
BITU	4	123 CT Road	9087123456	hike@gmail.com	06/03/01	21	60	9081234567	-
ASIM	2	Kharagpur	6294567800	asim@gmail.com	12/12/02	20	80	987123400	-

UPDATE

update student2 set marks=85 where name='RAJAT'; update student2 set
department='M.SC.(PG)' where roll=05;
update student2 set department='M.TECH.(PG)' where roll=06; update student2
set department='MBA(PG)' where roll=02; update student2 set
department='B.SC.(UG)' where roll=04; update student2 set
department='MCA(PG)' where roll=08;

DELETE

NAME	ROLL	ADDR	PHN_NO	EMAIL	DOB	AGE	MARKS	AADHAR_NO	DEPARTMENT
KRISHNENDU	6	Midnapore	9064557423	c25@gmail.com	30/04/01	21	90	1234567890	M.TECH(PG)
RAJAT	8	Kharagpur	9876543210	rajat@gmail.com	07/05/98	24	85	987654321	MCA(PG)
GANESH	5	Chondrokona	8798654312	yahoo@gmail.com	09/06/99	23	70	6789054321	M.SC.(PG)
BITU	4	123 CT Road	9087123456	hike@gmail.com	06/03/01	21	60	9081234567	B.SC.(UG)
ASIM	2	Kharagpur	6294567800	asim@gmail.com	12/12/02	20	80	987123400	MBA(PG)

delete from student2 where roll=3;

CONDITION (and, or, where)

select * from student2 where age>=22 and age<=25;

NAME	ROLL	ADDR	PHN_NO	EMAIL	DOB	AGE	MARKS	AADHAR_NO	DEPARTMENT
RAJAT	8	Kharagpur	9876543210	rajat@gmail.com	07/05/98	24	85	987654321	MCA(PG)
GANESH	5	Chondrokona	8798654312	yahoo@gmail.com	09/06/99	23	70	6789054321	M.SC.(PG)

NAME	ROLL	ADDR	PHN_NO	EMAIL	DOB	AGE	MARKS	AADHAR_NO	DEPARTMENT
KRISHNENDU	6	Midnapore	9064557423	c25@gmail.com	30/04/01	21	90	1234567890	M.TECH(PG)
RAJAT	8	Kharagpur	9876543210	rajat@gmail.com	07/05/98	24	85	987654321	MCA(PG)
GANESH	5	Chondrokona	8798654312	yahoo@gmail.com	09/06/99	23	70	6789054321	M.SC.(PG)
BITU	4	123 CT Road	9087123456	hike@gmail.com	06/03/01	21	60	9081234567	B.SC.(UG)

select * from student2 where marks=70 or marks=90;

NAME	ROLL	ADDR	PHN_NO	EMAIL	DOB	AGE	MARKS	AADHAR_NO	DEPARTMENT
KRISHNENDU	6	Midnapore	9064557423	c25@gmail.com	30/04/01	21	90	1234567890	M.TECH(PG)
GANESH	5	Chondrokona	8798654312	yahoo@gmail.com	09/06/99	23	70	6789054321	M.SC.(PG)

2. **Implementation of student table by using (ascending, descending, group by, order by, distinct, count, max, min, average, character recognition)**

ASCENDIING ORDER BY ROLL

select * from student2 order by roll asc;

NAME	ROLL	ADDR	PHN_NO	EMAIL	DOB	AGE	MARKS	AADHAR_NO	DEPARTMENT
BITU	4	123 CT Road	9087123456	hike@gmail.com	06/03/01	21	60	9081234567	B.SC.(UG)
GANESH	5	Chondrokona	8798654312	yahoo@gmail.com	09/06/99	23	70	6789054321	M.SC.(PG)
RAJAT	8	Kharagpur	9876543210	rajat@gmail.com	07/05/98	24	85	987654321	MCA(PG)
KRISHNENDU	6	Midnapore	9064557423	c25@gmail.com	30/04/01	21	90	1234567890	M.TECH(PG)

DESCENDIING ORDER BY MARKS

select * from student2 order by marks desc;

NAME	ROLL	ADDR	PHN_NO	EMAIL	DOB	AGE	MARKS	AADHAR_NO	DEPARTMENT
KRISHNENDU	6	Midnapore	9064557423	c25@gmail.com	30/04/01	21	90	1234567890	M.TECH(PG)
RAJAT	8	Kharagpur	9876543210	rajat@gmail.com	07/05/98	24	85	987654321	MCA(PG)
GANESH	5	Chondrokona	8798654312	yahoo@gmail.com	09/06/99	23	70	6789054321	M.SC.(PG)
BITU	4	123 CT Road	9087123456	hike@gmail.com	06/03/01	21	60	9081234567	B.SC.(UG)

GROUP BY DEPARTMENT

select department from student2 group by department;

DEPARTMENT
MCA(PG)
M.TECH(PG)
B.SC.(UG)
M.SC.(PG)

ORDER BY AGE

select roll, name, addr, age from student2 order by age;

ROLL	NAME	ADDR	AGE
6	KRISHNENDU	Midnapore	21
4	BITU	123 CT Road	21
5	GANESH	Chondrokona	23
8	RAJAT	Kharagpur	24

DISTINCT ADDRESS OF STUDENTS

select distinct(addr), name, phn_no from student2;

ADDR	NAME	PHN_NO
Midnapore	KRISHNENDU	9064557423
123 CT Road	BITU	9087123456
Kharagpur	RAJAT	9876543210
Chondrokona	GANESH	8798654312

COUNT NAME WHOES AGE IS LESS THAN 23

select count(name) from student2 where age<23;

COUNT(NAME)
2

COUNT MAXIMUM MARKS

select max(marks) from student2;

MAX(MARKS)
90

COUNT MINIMUM MARKS

select min(marks) from student2;

MAX(MARKS)
90

COUNT AVERAGE MARKS

select avg(marks) from student2;

AVG(MARKS)
76.25

CHARACTER RECOGNIZATIOIN

- **LAST CHARACTER**

select * from student2 where name like '%T';

NAME	ROLL	ADDR	PHN_NO	EMAIL	DOB	AGE	MARKS	AADHAR_NO	DEPARTMENT
RAJAT	8	Kharagpur	9876543210	rajat@gmail.com	07/05/98	24	85	987654321	MCA(PG)

- **FIRST CHARACTER**

select * from student2 where name like 'K%';

NAME	ROLL	ADDR	PHN_NO	EMAIL	DOB	AGE	MARKS	AADHAR_NO	DEPARTMENT
KRISHNENDU	6	Midnapore	9064557423	c25@gmail.com	30/04/01	21	90	1234567890	M.TECH(PG)

3. **Creating Employee table with various queries**
   - **i) Create a table employee (e_id, f_name, l_name, address, phone_no, email, salary, age, gender).**
   - **ii) Insert 5 rows into this table of employee.**
   - **iii) View the employee table. Create:**

create table employee2(e_id varchar(3),f_name varchar(15),l_name varchar(15),address varchar(25),phone_no numeric(10),email varchar(25),salary numeric(10),age numeric(3),gender varchar(8));

Insert:

insert into employee2 values(101,'Ganesh','Santra','Chondrokona',8976543209,'ganesh@gmail.com',20000,24,'Male'); insert into employee2 values(103,'Rajat','Santra','Chondrokona',8768905223,'rajat@gmail.com',50000,

20,'Male'); insert into employee2
values(105,'Krishnendu','Nanda','Medinipur',9064557423,'nanda@gmail.com',5
0000,21,'Male'); insert into employee2
values(102,'Lisa','Sanki','Kolkata',9876543312,'lisa@gmail.com',35000,18,'Fem
ale');
insert into employee2
values(104,'Bitu','Das','Kharagpur',9807654321,'bitu@gmail.com',30000,26,'Ma
le');

View:
select * from employee2;

E_ID	F_NAME	L_NAME	ADDRESS	PHONE_NO	EMAIL	SALARY	AGE	GENDER
101	Ganesh	Santra	Chondrokona	8976543209	ganesh@gmail.com	20000	24	Male
103	Rajat	Santra	Chondrokona	8768905223	rajat@gmail.com	50000	20	Male
105	Krishnendu	Nanda	Medinipur	9064557423	nanda@gmail.com	50000	21	Male
102	Lisa	Sanki	Kolkata	9876543312	lisa@gmail.com	35000	18	Female
104	Bitu	Das	Kharagpur	9807654321	bitu@gmail.com	30000	26	Male

iv) Add a column Department.
alter table employee2 add department varchar(15);

E_ID	F_NAME	L_NAME	ADDRESS	PHONE_NO	EMAIL	SALARY	AGE	GENDER	DEPARTMENT
101	Ganesh	Santra	Chondrokona	8976543209	ganesh@gmail.com	20000	24	Male	-
103	Rajat	Santra	Chondrokona	8768905223	rajat@gmail.com	50000	20	Male	-
105	Krishnendu	Nanda	Medinipur	9064557423	nanda@gmail.com	50000	21	Male	-
102	Lisa	Sanki	Kolkata	9876543312	lisa@gmail.com	35000	18	Female	-
104	Bitu	Das	Kharagpur	9807654321	bitu@gmail.com	30000	26	Male	-

v) Update Department details.
update employee2 set department='Management' where e_id=104; update
employee2 set department='Management' where e_id=103; update employee2
set department='Accounting' where e_id=105; update employee2 set
department='IT' where e_id=102;
update employee2 set department='IT' where e_id=101;

E_ID	F_NAME	L_NAME	ADDRESS	PHONE_NO	EMAIL	SALARY	AGE	GENDER	DEPARTMENT
101	Ganesh	Santra	Chondrokona	8976543209	ganesh@gmail.com	20000	24	Male	IT
103	Rajat	Santra	Chondrokona	8768905223	rajat@gmail.com	50000	20	Male	Management
105	Krishnendu	Nanda	Medinipur	9064557423	nanda@gmail.com	50000	21	Male	Accounting
102	Lisa	Sanki	Kolkata	9876543312	lisa@gmail.com	35000	18	Female	IT
104	Bitu	Das	Kharagpur	9807654321	bitu@gmail.com	30000	26	Male	Management

vi) Modify employee details, department = 'Accounting' whose id = 101.

update employee2 set department='Accounting' where e_id=101;

E_ID	F_NAME	L_NAME	ADDRESS	PHONE_NO	EMAIL	SALARY	AGE	GENDER	DEPARTMENT
101	Ganesh	Santra	Chondrokona	8976543209	ganesh@gmail.com	20000	24	Male	Accounting
103	Rajat	Santra	Chondrokona	8768905223	rajat@gmail.com	50000	20	Male	Management
105	Krishnendu	Nanda	Medinipur	9064557423	nanda@gmail.com	50000	21	Male	Accounting
102	Lisa	Sanki	Kolkata	9876543312	lisa@gmail.com	35000	18	Female	IT
104	Bitu	Das	Kharagpur	9807654321	bitu@gmail.com	30000	26	Male	Management

vii) Modify employee address whose id = 103 and 105.

update employee2 set address='Paschim Medinipur' where e_id=103 and e_id=105;

E_ID	F_NAME	L_NAME	ADDRESS	PHONE_NO	EMAIL	SALARY	AGE	GENDER	DEPARTMENT
101	Ganesh	Santra	Chondrokona	8976543209	ganesh@gmail.com	20000	24	Male	Accounting
103	Rajat	Santra	Paschim Medinipur	8768905223	rajat@gmail.com	50000	20	Male	Management
105	Krishnendu	Nanda	Paschim Medinipur	9064557423	nanda@gmail.com	50000	21	Male	Accounting
102	Lisa	Sanki	Kolkata	9876543312	lisa@gmail.com	35000	18	Female	IT
104	Bitu	Das	Kharagpur	9807654321	bitu@gmail.com	30000	26	Male	Management

viii) Find the employee details by their salary by ascending order.

select * from employee2 order by salary asc;

E_ID	F_NAME	L_NAME	ADDRESS	PHONE_NO	EMAIL	SALARY	AGE	GENDER	DEPARTMENT
101	Ganesh	Santra	Chondrokona	8976543209	ganesh@gmail.com	20000	24	Male	Accounting
104	Bitu	Das	Kharagpur	9807654321	bitu@gmail.com	30000	26	Male	Management
102	Lisa	Sanki	Kolkata	9876543312	lisa@gmail.com	35000	18	Female	IT
103	Rajat	Santra	Paschim Medinipur	8768905223	rajat@gmail.com	50000	20	Male	Management
105	Krishnendu	Nanda	Paschim Medinipur	9064557423	nanda@gmail.com	50000	21	Male	Accounting

ix) Find the employees details by their address.

select distinct(address),f_name,l_name,phone_no,email,age,salary,gender from employee2;

ADDRESS	F_NAME	L_NAME	PHONE_NO	EMAIL	AGE	SALARY	GENDER
Kolkata	Lisa	Sanki	9876543312	lisa@gmail.com	18	35000	Female
Paschim Medinipur	Rajat	Santra	8768905223	rajat@gmail.com	20	50000	Male
Kharagpur	Bitu	Das	9807654321	bitu@gmail.com	26	30000	Male
Chondrokona	Ganesh	Santra	8976543209	ganesh@gmail.com	24	20000	Male
Paschim Medinipur	Krishnendu	Nanda	9064557423	nanda@gmail.com	21	50000	Male

x) Count the employee's name whose salary > 20,000.

select count(f_name) from employee2 where salary>20000;

COUNT(F_NAME)
4

xi) Calculate the total salary of all employees.

select sum(salary) from employee2;

SUM(SALARY)
185000

xii) Count salary, name of employees group by their name.

select count(salary),f_name from employee2 group by f_name;

COUNT(SALARY)	F_NAME
1	Bitu
1	Krishnendu
1	Rajat
1	Ganesh
1	Lisa

4. **Joining two tables using cross join, inner join, left join, right join**

Create two tables name movie(movie_id,title,category) and another is member(member_id,name, movie_id)

----- Movie Table -----

Create:

create table movie(movie_id numeric(2),title varchar(15),category varchar(15));

Insert:

insert into movie values(1,'Pushpa','Action'); insert into movie values(2,'The Monk','Horror'); insert into movie values(3,'KGF','Action'); insert into movie values(4,'movie 3','Thriller');
insert into movie values(5,'Conjuring 3','Horror'); insert into movie values(6,",");

select * from movie;

View:

MOVIE_ID	TITLE	CATEGORY
1	Pushpa	Action
3	KGF	Action
4	movie 3	Thriller
5	Conjuring 3	Horror
6	-	-
2	The Monk	Horror

-----Member Table -----

Create:

create table member(member_id numeric(2),name varchar(15),movie_id numeric(2));

Insert:

insert into member values(34,'krishnendu',3); insert into member values(78,'',2); insert into member values(45,'bitu',1); insert into member values(23,'ashim',4); insert into member values(12,'rajat',5); insert into member values(90,'Arpan',6);

View:

select * from member;

MEMBER_ID	NAME	MOVIE_ID
34	krishnendu	3
45	bitu	1
78	-	2
12	rajat	5
90	Arpan	6
23	ashim	4

------ Cross Join -----

select * from movie cross join member;

MOVIE_ID	TITLE	CATEGORY	MEMBER_ID	NAME	MOVIE_ID
1	Pushpa	Action	34	krishnendu	3
1	Pushpa	Action	45	bitu	1
1	Pushpa	Action	78	-	2
1	Pushpa	Action	12	rajat	5
1	Pushpa	Action	90	Arpan	6
1	Pushpa	Action	23	ashim	4
3	KGF	Action	34	krishnendu	3
3	KGF	Action	45	bitu	1
3	KGF	Action	78	-	2
3	KGF	Action	12	rajat	5
More than 10 rows available. Increase rows selector to view more rows.					

------ Inner Join -----

select member.name,movie.title,movie.category from member,movie where member.movie_id=movie.movie_id;

NAME	TITLE	CATEGORY
bitu	Pushpa	Action
krishnendu	KGF	Action
ashim	movie 3	Thriller
rajat	Conjuring 3	Horror
Arpan	-	-
-	The Monk	Horror

------ Left Join -----

select movie.title,movie.category,member.name from movie left join member on movie.movie_id=member.movie_id;

TITLE	CATEGORY	NAME
KGF	Action	krishnendu
Pushpa	Action	bitu
The Monk	Horror	-
Conjuring 3	Horror	rajat
-	-	Arpan
movie 3	Thriller	ashim

------ Right Join -----

select member.name,movie.title,movie.category from member right join movie on member.movie_id = movie.movie_id;

NAME	TITLE	CATEGORY
krishnendu	KGF	Action
bitu	Pushpa	Action
-	The Monk	Horror
rajat	Conjuring 3	Horror
Arpan	-	-
ashim	movie 3	Thriller

5. **Creating three tables sailors, boat and reserve joining these tables with various query**

   i) **Create three tables name sailor(s_id,s_name,s_rating,s_age), boat(b_id,b_name,color,s_id) and reserve(s_id,b_id,hire_date)**

Inserting 5 rows into sailor, boat and reserve tables View these tables

---- SAILOR ----

Create:

create table sailor(s_id numeric(2),s_name varchar(25),s_rating numeric(10),s_age numeric(3));

Insert:

insert into sailor values(01,'Ganesh',5,40); insert into sailor values(02,'Ashim',2,30); insert into sailor values(03,'Bitu',4,50); insert into sailor values(04,'Rajat',3,70);
insert into sailor values(05,'Krishnendu',5,60); insert into sailor values(06,'Arpan',5,22);

select * from sailor;

View:

S_ID	S_NAME	S_RATING	S_AGE
1	Ganesh	5	40
2	Ashim	2	30
3	Bitu	4	50
4	Rajat	3	70
5	Krishnendu	5	60
6	Arpan	5	22

---- BOAT ----

Create:

create table boat(b_id numeric(5),s_id numeric(2),b_name varchar(20),color varchar(10));

Insert:

insert into boat values(101,03,'Sonar Tori','green'); insert into boat values(102,02,'Ovinondon','black'); insert into boat values(103,01,'Express','red');
insert into boat values(104,05,'Happy Journey','blue');

View:

select * from boat;

B_ID	S_ID	B_NAME	COLOR
101	3	Sonar Tori	green
102	2	Ovinondon	black
103	1	Express	red
104	5	Happy Journey	blue

--- RESERVE ---

Create:

create table reserve(s_id numeric(2),b_id numeric(5),hire_date varchar(20));

Insert:

insert into reserve values(04,103,'10/05/22'); insert into reserve values(05,101,'12/05/22'); insert into reserve values(01,103,'9/05/22'); insert into reserve values(05,103,'9/05/22'); insert into reserve values(04,102,'10/05/22');

select * from reserve;* from boat;

view:

S_ID	B_ID	HIRE_DATE
4	103	10/05/22
5	101	12/05/22
1	103	8/05/22
5	103	9/05/22
4	102	10/05/22

i) Find all information of sailor who have reserve boat number 101.

select * from sailor join boat on sailor.s_id=boat.s_id where b_id = 101;

S_ID	S_NAME	S_RATING	S_AGE	B_ID	S_ID	B_NAME	COLOR
3	Bitu	4	50	101	3	Sonar Tori	green

ii) Find the name of boat reserve by sailor name.

select boat.b_name,sailor.s_name from boat join sailor on boat.s_id=sailor.s_id where s_name='Krishnendu';

B_NAME	S_NAME
Happy Journey	Krishnendu

iii) Find the name of the sailor who have reserve a red boat and list in the order of age.

select sailor.s_name,boat.color from sailor join boat on sailor.s_id=boat.s_id where boat.color='red' order by sailor.s_age asc;

S_NAME	COLOR
Ganesh	red

iv) Find the name of sailor who have reserved at least one boat.

select sailor.s_name,reserve.b_id from sailor join reserve on sailor.s_id=reserve.s_id;

S_NAME	B_ID
Ganesh	103
Rajat	102
Rajat	103
Krishnendu	103
Krishnendu	101

v) **Find the name of sailor who have reserved difference boat on the same date.**

```
select sailor.s_name,reserve.b_id,reserve.hire_date from sailor
join reserve on sailor.s_id=reserve.s_id;
```

S_NAME	B_ID	HIRE_DATE
Ganesh	103	9/05/22
Rajat	102	10/05/22
Rajat	103	10/05/22
Krishnendu	103	9/05/22
Krishnendu	101	12/05/22

vi) **Find the id of sailor who have reserved a red boat or green boat.**

```
select sailor.s_id,boat.color from sailor join boat on
sailor.s_id=boat.s_id where boat.color='red' or boat.color='green';
```

S_ID	COLOR
1	red
3	green

vii) **Find the name and age of youngest sailor.**

```
select sailor.s_name,sailor.s_age from sailor where
sailor.s_age>=18 and sailor.s_age<=25;
```

S_NAME	S_AGE
Arpan	22

viii) **Count the number of different sailor name.**

```
select count(sailor.s_name) from sailor;
```

COUNT(SAILOR.S_NAME)
6

ix) **Find the average age of sailor for each rating levels.**

```
select avg(s_age) from sailor order by s_rating;
```

AVG(S_AGE)
45.333333333333333333333333333333333333

x) **Find the average age of sailor for each rating levels that at least two sailor.**

select avg(s_age) from sailor order by s_rating;

AVG(S_AGE)
45.333333333333333333333333333333333333

6. **Creating four tables movie, actor, acts and director joining these tables with various query.**

i) **Create four tables name movie (movie_id,title,dir_name,release_year,rating), actor(act_id,act_name, movie_id), acts(act_name,movie_name) and director(dir_name,movie_id,year)**

Inserting 5 rows into movie,actor,acts and director tables View these tables

--- MOVIE ---

Create:

create table movie1(movie_id numeric(5),title varchar(20),dir_name varchar(25),release_year numeric(6),rating numeric(3));

Insert:

insert into movie1 values(101,'Ramayan','Rahman',2018,5);
insert into movie1 values(102,'Mahavarat','Prashant Neel',2020,4); insert into movie1 values(103,'Story of Jungle','Tanveer Evan',2022,3); insert into movie1 values(104,'Super 30','Baba Yadab',2020,4);
insert into movie1 values(105,'Ravvan','Jeet Ganguly',2021,2); insert into movie1 values(106,'KGF','Prashant Neel',2022,5);
insert into movie1 values(107,'Spider Man','Prashant Neel',2021,4);

View:

select * from movie1;

MOVIE_ID	TITLE	DIR_NAME	RELEASE_YEAR	RATING
101	Ramayan	Rahman	2018	5
102	Mahavarat	Prashant Neel	2020	4
103	Story of Jungle	Tanveer Evan	2022	3
104	Super 30	Baba Yadab	2020	4
105	Ravvan	Jeet Ganguly	2021	2
106	KGF	Prashant Neel	2022	5
107	Spider Man	Prashant Neel	2021	4

--- ACTOR ---

Create:

create table actor(act_id numeric(5),act_name varchar(25),movie_id numeric(5));

Insert:

insert into actor values(001,'Jeet',105); insert into actor values(002,'Yash',102); insert into actor values(003,'Tiger Shrof',103); insert into actor values(004,'Hrittik',104); insert into actor values(005,'Amir Khan',101); insert into actor values(006,'Yash',106);
insert into actor values(007,'Bidyut Jamal',107);

View:

select * from actor;

ACT_ID	ACT_NAME	MOVIE_ID
1	Jeet	105
2	Yash	102
3	Tiger Shrof	103
4	Hrittik	104
5	Amir Khan	101
6	Yash	106
7	Bidyut Jamal	107

--- ACTS ---

Create:

create table acts(act_name varchar(25),movie_name varchar(20));

View:

Insert:

insert into acts values('Amir Khan','Ramayan'); insert into acts values('Yash','KGF');
insert into acts values('Yash','Mahavarat'); insert into acts values('Jeet','Ravvan');

select * from acts;

ACT_NAME	MOVIE_NAME
Jeet	Ravvan
Yash	Mahavarat
Yash	KGF
Amir Khan	Ramayan

--- DIRECTOR ---

Create:

create table director1(dir_name varchar(25),movie_id numeric(5),year numeric(6));

Insert:

insert into director1 values('Prashant Neel',106,2022); insert into director1 values('Tanveer Evan',103,2022); insert into director1 values('Rahman',101,2018); insert into director1 values('Prashant Neel',102,2020); insert into director1 values('Prashant Neel',107,2021);

View:

select * from director1;

DIR_NAME	MOVIE_ID	YEAR
Prashant Neel	106	2022
Tanveer Evan	103	2022
Rahman	101	2018
Prashant Neel	102	2020
Prashant Neel	107	2021

ii) Find the movie name made after 2020 acts by 'Yash'.

select movie1.title,actor.act_name,movie1.release_year from movie1 join actor on movie1.movie_id=actor.movie_id where movie1.release_year>2020 and actor.act_name='Yash';

TITLE	ACT_NAME	RELEASE_YEAR
KGF	Yash	2022

iii) Find the movies made by 'Prasanth Neel' in the year 2018 which is rating above 4 stars.

select movie1.title,movie1.release_year,movie1.rating,director1.dir_name from movie1 join director1 on movie1.movie_id=director1.movie_id where movie1.release_year=2018 and movie1.rating>4 and director1.dir_name='Rahman';

TITLE	RELEASE_YEAR	RATING	DIR_NAME
Ramayan	2018	5	Rahman

iv) Find all movies with their ratings in ascending order.

select movie1.title, movie1.rating from movie1 order by movie1.rating asc;

TITLE	RATING
Ravvan	2
Story of Jungle	3
Spider Man	4
Mahavarat	4
Super 30	4
KGF	5
Ramayan	5

v) Find movies name which is directed by 'Prashant Neel' and doesn't acts by 'Yash'.

select movie1.title,movie1.dir_name,actor.act_name from movie1 join actor on movie1.movie_id=actor.movie_id where movie1.dir_name='Prashant Neel' and actor.act_name!='Yash';

TITLE	DIR_NAME	ACT_NAME
Spider Man	Prashant Neel	Bidyut Jamal

vi) Find all actor and director name who are combine the same movie in the year 2022.

select movie1.dir_name, movie1.release_year, actor.act_name from movie1 join actor on movie1.movie_id= actor.movie_id where movie1.release_year=2022;

DIR_NAME	RELEASE_YEAR	ACT_NAME
Tanveer Evan	2022	Tiger Shrof
Prashant Neel	2022	Yash

7. **SQL queries of find the System date and time.**

i) How to find system date from database.

select SYSDATE from dual;

SYSDATE
09-JUN-22

ii) How to find system time from database.

select SYSTIMESTAMP from dual;

SYSTIMESTAMP
09-JUN-22 09.58.37.811000 AM +05:30

iii) How to display 1 to 100 number with using query.

select level from dual connect by level<=100

LEVEL
1
2
3
4
5
6
7
8
9
10
More than 10 rows available. Increase rows selector to view more rows.

8. **Creating six tables book, author, publisher, book-copies, book-lending and library-branch and joining these tables with various queries.**

i) **Create four tables name – book(b_id,title,pub_name,pub_year), author(a_id,b_id,a_name), publisher(pub_id,pub_name,address,phone_no), bookcopies(b_id,branch_id,number_of_copies), booklending(b_id,branch_id,card_no,purchase_date), librarybranch(branch_id,branch_name,address)**
Inserting 5 rows into these tables View these tables

--- BOOK ---
Create:
create table book(b_id numeric(3),title varchar(20),pub_name varchar(15),pub_year numeric(4));
Insert:
insert into book values(101,'Physics','Santra Pub.',2022); insert into book values(102,'Math','Ray & Martin',2018); insert into book values(103,'Chemistry','Parul Pro.',2019); insert into book values(104,'Biology','Chaya Prokasoni',2021);
insert into book values(105,'Geography','Goutam Mallick',2022);
View:
select * from book;

B_ID	TITLE	PUB_NAME	PUB_YEAR
101	Physics	Santra Pub.	2022
102	Math	Ray & Martin	2018
103	Chemistry	Parul Pro.	2019
104	Biology	Chaya Prokasoni	2021
105	Geography	Goutam Mallick	2022

--- AUTHOR ---
Create:
create table author(a_id numeric(3),b_id numeric(3),a_name varchar(25));
Insert:
insert into author values(201,103,'Sukhendu Maity'); insert into author values(202,101,'R. Dey');
insert into author values(203,102,'Sourendranath Dey'); insert into author values(204,105,'Jack');
insert into author values(205,104,'Bhunia & Dhor');

View:
select * from author;

A_ID	B_ID	A_NAME
201	103	Sukhendu Maity
202	101	R. Dey
203	102	Sourendranath Dey
204	105	Jack
205	104	Bhunia & Dhor

--- PUBLISHER ---
Create:
create table publisher(pub_id varchar(5),pub_name varchar(20),address varchar(25),phone_no numeric(10));
Insert:
insert into publisher values('P123','Chaya Prokasoni','Kolkata',9087654321);
insert into publisher values('P124','Ray & Martin','Holdia',8790654321); insert into publisher values('P125','Santra Pub.','Mumbai',0987123456); insert into publisher values('P126','Parul Pro.','Gujrat',6543127890);
insert into publisher values('P127','Ray','Midnapore',1234097667);
View:
select * from publisher;

PUB_ID	PUB_NAME	ADDRESS	PHONE_NO
P123	Chaya Prokasoni	Kolkata	9087654321
P124	Ray & Martin	Holdia	8790654321
P125	Santra Pub.	Mumbai	987123456
P126	Parul Pro.	Gujrat	6543127890
P127	Ray	Midnapore	1234097667

--- BOOKCOPIES ---
Create:
create table bookcopies(b_id numeric(3),branch_id varchar(4),num_of_copies numeric(5));
Insert:
insert into bookcopies values(103,'B001',100); insert into bookcopies values(105,'B002',800); insert into bookcopies values(102,'B003',450); insert into bookcopies values(101,'B004',1000); insert into bookcopies values(104,'B005',500);

View:
select * from bookcopies;

B_ID	BRANCH_ID	NUM_OF_COPIES
103	B001	100
105	B002	800
102	B003	450
101	B004	1000
104	B005	500

--- BOOKLENDING ---
Create:
create table booklending(b_id numeric(3),branch_id varchar(4),card_no varchar(5),purchase_date varchar(12));
Insert:
insert into booklending values(102,'B003','C0vT5','21/02/2022'); insert into booklending values(105,'B002','C87Tb','08/05/2022'); insert into booklending values(104,'B005','C34y3','01/06/2022'); **View:**
select * from booklending;

B_ID	BRANCH_ID	CARD_NO	PURCHASE_DATE
102	B003	C0vT5	21/02/2022
102	B003	C0vT5	21/02/2022
105	B002	C87Tb	08/05/2022
104	B005	C34y3	01/06/2022

--- LIBRARY BRANCH ---
Create:
create table librarybranch(branch_id varchar(4),branch_name varchar(25),address varchar(25));
Insert:
insert into librarybranch values('B004','Physics Branch','Mumbai'); insert into librarybranch values('B002','English Branch','Midnapore'); insert into librarybranch values('B001','Chemistry Branch','Gujrat'); insert into librarybranch values('B005','Biology Branch','Kolkata'); insert into librarybranch values('B003','Mathematics Branch','Holdia');

View:

select * from librarybranch;

BRANCH_ID	BRANCH_NAME	ADDRESS
B004	Physics Branch	Mumbai
B004	Physics Branch	Mumbai
B002	English Branch	Midnapore
B001	Chemistry Branch	Gujrat
B005	Biology Branch	Kolkata
B003	Mathematics Branch	Holdia

ii) Find the details of all books in the library with their id, title, name of publisher, author and number of copies in each branch.

select book.b_id,book.title,book.pub_name,bookcopies.num_of_copies from book join bookcopies on book.b_id= bookcopies.b_id;

B_ID	TITLE	PUB_NAME	NUM_OF_COPIES
103	Chemistry	Parul Pro.	100
105	Computer	Ray	800
105	English	Ray	800
102	Math	Ray & Martin	450
101	Physics	Santra Pub.	1000
104	Biology	Chaya Prokasoni	500

iii) Get the particular borrower who have borrow more than 3 books from 27th january to 27th june.

select book.b_id,book.title,book.pub_name,bookcopies.num_of_copies from book join bookcopies on book.b_id= bookcopies.b_id;

BRANCH_ID	NUM_OF_COPIES	PURCHASE_DATE
B002	800	08/05/2022
B003	450	21/02/2022
B003	450	21/02/2022
B005	500	01/06/2022

iv) Delete a book name from the book table and update this

delete name from book where title=Geography; update book set title= 'History' where b_id=105;

B_ID	TITLE	PUB_NAME	PUB_YEAR
101	Physics	Santra Pub.	2022
102	Math	Ray & Martin	2018
103	Chemistry	Parul Pro.	2019
104	Biology	Chaya Prokasoni	2021
105	History	Ray	2022

v) Partition the book table based on year of publication

select book.title,pub_year from book;

TITLE	PUB_YEAR
Physics	2022
Math	2018
Chemistry	2019
Biology	2021
History	2022

vi) Create a view of all books and its number of copies that are currently available in the library

select book.title,bookcopies.num_of_copies from book join bookcopies on book.b_id=bookcopies.b_id;

TITLE	NUM_OF_COPIES
Chemistry	100
History	800
Math	450
Physics	1000
Biology	500

9. **Creating three tables Salesman, Customers and orders and joining these tables with various queries.**

i) Create four tables name salesman(s_id,s_same,s_city,commission),customers(c_id,c_name,c_city, grade,s_id), orders(order_num,purchase_amount,order_date,c_id,s_id)
Inserting 5 rows into these tables and view these tables
--- SALESMAN ---
Create:
create table salesmans(s_id numeric(3),s_name varchar(25),s_city varchar(30),commission numeric(5));
Insert:
insert into salesmans values(101,'Ashim Sarkar','Kolkata',2000); insert into salesmans values(102,'Ganesh Santra','Midnapore',1500); insert into salesmans values(103,'Rajat Santra','Chondrokona',3000); insert into salesmans values(104,'Bitu Das','Ghatal',4000);
insert into salesmans values(105,'Krishnendu Nanda','Midnapore',5000);
View:
select * from salesmans;

S_ID	S_NAME	S_CITY	COMMISSION
101	Ashim Sarkar	Kolkata	2000
102	Ganesh Santra	Midnapore	1500
103	Rajat Santra	Chondrokona	3000
104	Bitu Das	Ghatal	4000
105	Krishnendu Nanda	Midnapore	5000

--- CUSTOMERS ---
Create:
create table customers(c_id varchar(3),c_name varchar(20),c_city varchar(25),grade numeric(2),s_id numeric(3));
Insert:
insert into customers values('C1','Tamal Ghosh','Mumbai',3,102); insert into customers values('C2','Rittwik Jana','Gujrat',4,103); insert into customers values('C3','Robiul Islam','Kolkata',5,104); insert into customers values('C4','Akash bera','Midnapore',4,101); insert into customers values('C5','Ritam Dash','Durgapore',5,105); insert into customers values('C2','Jayanta Barik','Bangalore',5,103); insert into customers values('C3','Ayan Kamila','Bangalore',3,102); insert into customers values('C3','Robiul Islam','Kolkata',4,101);

View:
select * from customers;

C_ID	C_NAME	C_CITY	GRADE	S_ID
C1	Tamal Ghosh	Mumbai	3	102
C2	Rittwik Jana	Gujrat	4	103
C3	Robiul Islam	Kolkata	5	104
C4	Akash bera	Midnapore	4	101
C5	Ritam Dash	Durgapore	5	105
C2	Jayanta Barik	Bangalore	5	103
C3	Ayan Kamila	Bangalore	3	102
C3	Robiul Islam	Kolkata	4	101
C3	Jayanta Barik	Bangalore	5	103

--- ORDERS ---
Create:
create table orders(order_num varchar(5),purchase_amount numeric(5),order_date varchar(10),c_id varchar(3),s_id numeric(3));
Insert:
insert into orders values('OR123',300,'20/05/22','C2',105); insert into orders values('OR456',500,'15/04/22','C1',103); insert into orders values('OR789',1000,'30/03/22','C3',102); insert into orders values('OR012',2500,'05/06/22','C4',104); insert into orders values('OR001',3500,'18/06/22','C5',101); insert into orders values('OR345',700,'18/06/22','C5',101); insert into orders values('OR980',1500,'18/06/22','C5',101); **View:**
select * from orders;

ORDER_NUM	PURCHASE_AMOUNT	ORDER_DATE	C_ID	S_ID
OR123	300	20/05/22	C2	105
OR456	500	15/04/22	C1	103
OR789	1000	30/03/22	C3	102
OR012	2500	05/06/22	C4	104
OR001	3500	18/06/22	C5	101
OR345	700	18/06/22	C5	101
OR980	1500	18/06/22	C5	101

ii) Count the customers with their grades above the average who lives in 'Bangalore'.
select grade,count(distinct c_id) from customers group by grade having grade>(select avg(grade) from customers where c_city='Bangalore');

GRADE	COUNT(DISTINCTC_ID)
5	3

ORDER_NUM	ORDER_DATE	PURCHASE_AMOUNT	S_ID	S_NAME
OR345	18/06/22	700	101	Ashim Sarkar
OR980	18/06/22	1500	101	Ashim Sarkar
OR001	18/06/22	3500	101	Ashim Sarkar
OR789	30/03/22	1000	102	Ganesh Santra
OR456	15/04/22	500	103	Rajat Santra
OR012	05/06/22	2500	104	Bitu Das
OR123	20/05/22	300	105	Krishnendu Nanda

iii) Find the name and numbers of all salesman who have more than one customer.

select s_id,s_name from salesmans where 1<(select count(*) from customers where customers.s_id=salesmans.s_id);

S_ID	S_NAME
101	Ashim Sarkar
102	Ganesh Santra
103	Rajat Santra

iv) List all salesman and indicate those who have and don't have customer in their cities.

select salesmans.s_id,salesmans.s_name,salesmans.s_city,customers.c_id,customers.c_name,customers.c_city from salesmans join customers on salesmans.s_id=customers.s_id where salesmans.s_city=customers.c_city or salesmans.s_city!=customers.c_city;

S_ID	S_NAME	S_CITY	C_ID	C_NAME	C_CITY
102	Ganesh Santra	Midnapore	C1	Tamal Ghosh	Mumbai
103	Rajat Santra	Chondrokona	C2	Rittwik Jana	Gujrat
104	Bitu Das	Ghatal	C3	Robiul Islam	Kolkata
101	Ashim Sarkar	Kolkata	C4	Akash bera	Midnapore
105	Krishnendu Nanda	Midnapore	C5	Ritam Dash	Durgapore
103	Rajat Santra	Chondrokona	C2	Jayanta Barik	Bangalore
102	Ganesh Santra	Midnapore	C3	Ayan Kamila	Bangalore
101	Ashim Sarkar	Kolkata	C3	Robiul Islam	Kolkata
103	Rajat Santra	Chondrokona	C3	Jayanta Barik	Bangalore

v) Create a view that finds the salesman who has the customer with the highest order of a day.

select orders.order_num,orders.order_date,orders.purchase_amount,salesmans.s_id,salesmans.s_name from orders join salesmans on orders.s_id=salesmans.s_id order by orders.s_id asc;

vi) Delete the salesman with id '101'.

delete from salesmans where s_id=101;

S_ID	S_NAME	S_CITY	COMMISSION
102	Ganesh Santra	Midnapore	1500
103	Rajat Santra	Chondrokona	3000
104	Bitu Das	Ghatal	4000
105	Krishnendu Nanda	Midnapore	5000

BCAHMJ06P:
OPERATING SYSTEMS LABORATORY MANUAL

SL.NO.	Experiments
1.	Write a program (using fork () and/or exec () commands) where parent and child execute: a) same program, same code. b) same program, different code. c) before terminating, the parent waits for the child to finish its task.
2.	Write a program to report behaviour of Linux kernel including kernel version, CPU type and model. (CPU information)
3.	Write a program to report behaviour of Linux kernel including information on configured memory, amount of free and used memory (memory information).

4.	Write a program to print file details including owner access permissions, file access time, where file name is given as argument.
5.	Write a program to copy files using system calls.
6.	Write program to implement FCFS scheduling algorithm.
7.	Write program to implement Round Robin scheduling algorithm.
8.	Write program to implement SJF scheduling algorithm.
9.	Write program to calculate sum of n numbers using thread library.
10.	Write a program to implement first-fit, best-fit and worst-fit allocation strategies

1. Write a program (using fork () and/or exec () commands) where parent and child execute:

a) same program, same code.

Program:

sanjoy@SANJUVAI:~$ gedit same.c

```
#include<stdio.h>
#include<unistd.h>
#include<stdlib.h>
int main(){
int pid_t,pid,p;
p=fork();
```

```
pid=getpid();
if(p<0){
        fprintf(stderr,"Frok failded");
return 1;
}
printf("Output of fork id: %d \n",p);
printf("process id is : %d \n",pid);
return 0;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ gcc same.c -o sh.out
sanjoy@SANJUVAI:~$ ./sh.out
Output of fork id: 564
process id is : 563
Output of fork id: 0
process id is : 564
sanjoy@SANJUVAI:~$
```

b) same program, different code.

Program:

```
sanjoy@SANJUVAI:~$ gedit different.c
#include<stdio.h>
#include<unistd.h>
#include<stdlib.h>
int main(){
int pid;
pid=fork();
if(pid<0){
printf("\n Error ");
exit(1);
}
 else if(pid==0){
   printf("\n Hello I am child process ");
   printf("\n my pid is %d \n",getpid());
   exit(0);
 }
  else{
  printf("\n Hello i am parent process  ");
  printf("\n My actual pid is %d \n",getpid());
  exit(1);
```

```
  }
return 0;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ gcc different.c -o sh.out
sanjoy@SANJUVAI:~$ ./sh.out

 Hello i am parent process
 My actual pid is 714

 Hello I am child process
 my pid is 715
```

c) before terminating, the parent waits for the child to finish its task.

Program:

```
sanjoy@SANJUVAI:~$ gedit parentwait.c
#include<stdio.h>
#include<unistd.h>
#include<stdlib.h>
int main(){
int pid;
pid=fork();
if(pid<0){
printf("\n Error ");
exit(1);
}
 else if(pid==0){
   printf("\n Hello I am child process ");
   printf("\n my pid is %d \n",getpid());
   exit(0);
 }
  else if(pid>0){
  printf("\n Hello i am parent process  ");
  printf("\n My actual pid is %d \n",getpid());
  //wait(NULL);
  exit(1);
  }
return 0;
}
```

Input and Output Section:


```
sanjoy@SANJUVAI:~$ gcc parentwait.c -o sh.out
```

```
sanjoy@SANJUVAI:~$ ./sh.out

 Hello i am parent process
 My actual pid is 907

 Hello I am child process
 my pid is 908
```

2. Write a program to report behaviour of Linux kernel including kernel version, CPU type and model. (CPU information)

Program:

```
#include<stdio.h>
#include<stdlib.h>
#include<sys/utsname.h>
using namespace std;
int main()
{
   int m=0;
   struct utsname s1;
   m=uname(&s1);

   if(m==0)
   {
      printf("\n The name of System:%s , system" , s1.sysname);
      printf("\n The version:%s" , s1.version);
      printf("\n The Machine:%s" , s1.machine);
      printf("\n");
      system("cat /proc/cpuinfo | awk 'NR==3, NR==4{print}' \n");
   }

   else
   {
      printf("Error");
   }
   return 0;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ touch question2.cpp
sanjoy@SANJUVAI:~$ g++ question2.cpp
sanjoy@SANJUVAI:~$ ./a.out

 The name of System:Linux , system
 The version:#1 SMP Wed Nov 23 01:01:46 UTC 2022
```

```
The Machine:x86_64
cpu family      : 6
model           : 126
```

3. ***Write a program to report behaviour of Linux kernel including information on configured memory, amount of free and used memory (memory information).***

Program:

```
#include<stdlib.h>
#include<iostream>
using namespace std;
int main()
{
cout<<"\nThe kernel version is, \n";
system("cat /proc/sys/kernel/osrelease") ;
cout<<"\nThe CPU space: \n";
system("cat /proc/cpuinfo | awk 'NR==3, NR==4{print}' \n ");
cout<<"\nAmount of cpu time since system was last booted is: ";
system("cat /proc/uptime \n");
cout<<"\nThe configured memory is : \n";
system("cat /proc/meminfo | awk 'NR==1, NR==4{print $2}' \n ");
//cout<<"\nAmount of free memory: \n";
//system("cat /proc/meminfo |awk 'NR = 2{Print $2}' \n ");
cout<<"Amount of used memory is: \n";
system("cat /proc/meminfo | awk '{ if (NR==1) a=$2; if(NR==2) b=$2 }END
        {print a-b} ' \n");
return 0;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ g++ memory.cpp
sanjoy@SANJUVAI:~$ ./a.out

The kernel version is,
5.15.79.1-microsoft-standard-WSL2

The CPU space:
cpu family      : 6
model           : 126

3531.89 28231.67
Amount of cpu time since system was last booted is:
The configured memory is :
```

```
3880936
3286912
3380688
11896
Amount of used memory is:
594024
```

4. Write a program to print file details including owner access permissions, file access time, where file name is given as argument.

Program:

```
#include<iostream>
using namespace std;
#include<stdlib.h>
#include<stdio.h>
#include<sys/stat.h>
#include<sys/types.h>
#include<unistd.h>
int main(int argc, char *argv[])
{
int i;
struct stat s;
if (argc < 2)
{
cout<<"\n enter filename in";
//exit();
}
for(i=1;i<argc; i++)
{
cout<<"File : "<<argv[i]<<"\n";
if(stat(argv[i],&s)<0)
cout<<"error in obtaining stats In";
else
{
cout<<"owner UID : "; cout<<s.st_uid; cout<<"\n";
cout<<"group ID :"; cout<<s.st_gid; cout<<"\n";
cout<<"Access permissions : "; cout<<s.st_mode; cout<<"\n";
cout<<"Access Time :" ;cout<<s.st_atime; cout<<"\n";
cout<<"File Size : "; cout<<s.st_size; cout<<"\n";
cout<<"File Size(in blocks) : "; cout<<s.st_blksize; cout<<"\n";
}
}
return 0;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ cat>f1
Well, Come to Midnapore City College.
^Z
[3]+  Stopped                 cat > f1
sanjoy@SANJUVAI:~$ g++ test2.cpp
sanjoy@SANJUVAI:~$ ./a.out f1
File : f1
owner UID : 1000
group ID :1000
Access permissions : 33188
Access Time :1674936817
File Size : 37
File Size(in blocks) : 4096
```

5. Write a program to copy files using system calls.

Program:

```
sanjoy@SANJUVAI:~$ gedit copyfile.c

#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
#include <stdlib.h>
int main(int argc, char *argv[])
 {
  int f1, f2;
  char buff[50];
  long int n;

 if(((f1 = open(argv[1], O_RDONLY)) == -1 || ((f2=open(argv[2], O_CREAT |
 O_WRONLY | O_TRUNC, 0700))== 1)))
   {
     perror("problem in file");
     exit(1);
   }

 while((n=read(f1, buff, 50))>0)

    if(write(f2, buff, n)!=n)
      {
        perror("problem in writing");
        exit(3);
```

```
      }

    if(n==-1)
      {
        perror("problem in reading");
        exit(2);
      }

     close(f2);
     exit(0);
}
```

Input and Output Section:

```
            sanjoy@SANJUVAI:~$ cc copyfile.c
sanjoy@SANJUVAI:~$ cat>bsc.txt
This is BSc Computer Science File.
BSc Computer Science File is copy into BCA File.
^Z
[7]+  Stopped                 cat > bsc.txt
sanjoy@SANJUVAI:~$ ./a.out bsc.txt bca.txt
sanjoy@SANJUVAI:~$ cat bca.txt
This is BSc Computer Science File.
BSc Computer Science File is copy into BCA File.
```

6. Write program to implement FCFS scheduling algorithm.

Program:

```
// C program for implementation of FCFS scheduling
#include<stdio.h>
// Function to find the waiting time for all processes
void findWaitingTime(int processes[], int n, int bt[], int wt[])
{
            // waiting time for first process is 0
            wt[0] = 0;

            // calculating waiting time
            for (int i = 1; i < n ; i++ )
                wt[i] = bt[i-1] + wt[i-1] ;
}

// Function to calculate turn around time
void findTurnAroundTime( int processes[], int n, int bt[], int wt[], int tat[])
{
```

```
        // calculating turnaround time by adding bt[i] + wt[i]
        for (int i = 0; i < n ; i++)
            tat[i] = bt[i] + wt[i];
}
//Function to calculate average time
void findavgTime( int processes[], int n, int bt[])
{
        int wt[n], tat[n], total_wt = 0, total_tat = 0;
        //Function to find waiting time of all processes
        findWaitingTime(processes, n, bt, wt);
        //Function to find turn around time for all processes
        findTurnAroundTime(processes, n, bt, wt, tat);
        //Display processes along with all details
        printf("Processes Burst time Waiting time Turn around time\n");
        // Calculate total waiting time and total turn around time
        for (int i=0; i<n; i++)
        {
            total_wt = total_wt + wt[i];
            total_tat = total_tat + tat[i];
            printf(" %d ",(i+1));
            printf("\t   %d", bt[i] );
            printf("\t\t %d",wt[i] );
            printf("\t\t %d\n",tat[i] );
        }
        int s=(float)total_wt / (float)n;
        int t=(float)total_tat / (float)n;
        printf("Average waiting time = %d",s);
        printf("\n");
        printf("Average turn around time = %d ",t);
}
int main()
{
        //process id's
        int processes[] = { 1, 2, 3};
        int n = sizeof processes / sizeof processes[0];

        //Burst time of all processes
        int burst_time[] = {10, 5, 8};

        findavgTime(processes, n, burst_time);
        return 0;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ gcc fcfs.c -o a.out
sanjoy@SANJUVAI:~$ ./a.out
Processes Burst time Waiting time Turn around time
 1        10          0            10
 2        5           10           15
 3        8           15           23
Average waiting time = 8
Average turn around time = 16
```

7. *Write program to implement Round Robin scheduling algorithm.*

Program:

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

// Use for loop to enter the details of the process like Arrival time and the Burst
            Time
for(i=0; i<NOP; i++)
{
printf("\n Enter the Arrival and Burst time of the Process[%d]\n", i+1);
printf(" Arrival time is: \t");  // Accept arrival time
scanf("%d", &at[i]);
printf(" \nBurst time is: \t"); // Accept the Burst time
scanf("%d", &bt[i]);
temp[i] = bt[i]; // store the burst time in temp array
}
// Accept the Time qunat
printf("Enter the Time Quantum for the process: \t");
scanf("%d", &quant);
// Display the process No, burst time, Turn Around Time and the waiting time
printf("\n Process No \t\t Burst Time \t\t TAT \t\t Waiting Time ");
for(sum=0, i = 0; y!=0; )
{
if(temp[i] <= quant && temp[i] > 0) // define the conditions
{
   sum = sum + temp[i];
```

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

Input and Output Section:

Total number of process in the system: 4

Enter the Arrival and Burst time of the Process[1]

Arrival time is: 0

```
Burst time is:  8

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

Burst time is:  5

 Enter the Arrival and Burst time of the Process[3]
 Arrival time is:       2

Burst time is:  10

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

Burst time is:  11
Enter the Time Quantum for the process:         6

 Process No          Burst Time          TAT          Waiting Time
Process No[2]         5                    10              5
Process No[1]         8                    25              17
Process No[3]         10                    27              17
Process No[4]         11                    31              20
 Average Turn Around Time:      14.750000
 Average Waiting Time:  23.250000
```

8. Write program to implement SJF scheduling algorithm.

Program:

```
sanjoy@SANJUVAI:~$ gedit sjf.cpp
#include <bits/stdc++.h>
using namespace std;
//structure for every process
struct Process {
   int pid; // Process ID
   int bt; // Burst Time
   int art; // Arrival Time
};
void findTurnAroundTime(Process proc[], int n, int wt[], int tat[]) {
   for (int i = 0; i < n; i++)
   tat[i] = proc[i].bt + wt[i];
}
//waiting time of all process
```

```
void findWaitingTime(Process proc[], int n, int wt[]) {
  int rt[n];
  for (int i = 0; i < n; i++)
  rt[i] = proc[i].bt;
  int complete = 0, t = 0, minm = INT_MAX;
  int shortest = 0, finish_time;
  bool check = false;
  while (complete != n) {
    for (int j = 0; j < n; j++) {
      if ((proc[j].art <= t) && (rt[j] < minm) && rt[j] > 0) {
        minm = rt[j];
        shortest = j;
        check = true;
      }
    }
    if (check == false) {
      t++;
      continue;
    }
    // decrementing the remaining time
    rt[shortest]--;
    minm = rt[shortest];
    if (minm == 0)
      minm = INT_MAX;
      // If a process gets completely
      // executed
      if (rt[shortest] == 0) {
        complete++;
        check = false;
        finish_time = t + 1;
        // Calculate waiting time
        wt[shortest] = finish_time -
        proc[shortest].bt -
        proc[shortest].art;
        if (wt[shortest] < 0)
          wt[shortest] = 0;
      }
      // Increment time
      t++;
  }
}
// Function to calculate average time
void findavgTime(Process proc[], int n) {
  int wt[n], tat[n], total_wt = 0,
```

```
  total_tat = 0;
  // Function to find waiting time of all
  // processes
  findWaitingTime(proc, n, wt);
  // Function to find turn around time for
  // all processes
  findTurnAroundTime(proc, n, wt, tat);
  cout << "Processes " << " Burst time " << " Waiting time " << " Turn around
          time\n";
  for (int i = 0; i < n; i++) {
    total_wt = total_wt + wt[i];
    total_tat = total_tat + tat[i];
    cout << " " << proc[i].pid << "\t\t" << proc[i].bt << "\t\t " << wt[i] << "\t\t "
          << tat[i] << endl;
  }
          cout << "\nAverage waiting time = " << (float)total_wt / (float)n;
cout << "\nAverage turn around time = " << (float)total_tat / (float)n;
}
int main() {
  Process proc[] = { { 1, 5, 1 }, { 2, 3, 1 }, { 3, 6, 2 }, { 4, 5, 3 } };
  int n = sizeof(proc) / sizeof(proc[0]);
  findavgTime(proc, n);
  return 0;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ g++ sjf.cpp
sanjoy@SANJUVAI:~$ ./a.out
Processes  Burst time  Waiting time  Turn around time
 1          5          3          8
 2          3          0          3
 3          6          12          18
 4          5          6          11

Average waiting time = 5.25
Average turn around time = 10
```

9. Write program to calculate sum of n numbers using thread library.

Program:

```
sanjoy@SANJUVAI:~$ gedit thread.c
#include <pthread.h>
#include <stdlib.h>
#include <stdio.h>
```

```
typedef struct data{
    int* arr;
    int thread_num;
} data;
int arrSize = 10;
void* halfSum(void* p){
    data* ptr = (data*)p;
    int n = ptr->thread_num;
    // Declare sum dynamically to return to join:
    int* thread_sum = (int*) calloc(1, sizeof(int));

    if(n == 0){
        for(int i = 0; i < arrSize/2; i++)
            thread_sum[0] = thread_sum[0] + ptr->arr[i];
    }
    else{
        for(int i = arrSize/2; i < arrSize; i++)
            thread_sum[0] = thread_sum[0] + ptr->arr[i];
    }

    pthread_exit(thread_sum);
}
int main(void){
    // Declare integer array [1,2,3,4,5,6,7,8,9,10]:
    int* int_arr = (int*) calloc(arrSize, sizeof(int));
    for(int i = 0; i < arrSize; i++)
        int_arr[i] = i + 1;
    // Declare arguments for both threads:
    data thread_data[2];
    thread_data[0].thread_num = 0;
    thread_data[0].arr = int_arr;
    thread_data[1].thread_num = 1;
    thread_data[1].arr = int_arr;
    // Declare thread IDs:
    pthread_t tid[2];
    // Start both threads:
    pthread_create(&tid[0], NULL, halfSum, &thread_data[0]);
    pthread_create(&tid[1], NULL, halfSum, &thread_data[1]);
    // Declare space for sum:
    int* sum0;
    int* sum1;
    // Retrieve sum of threads:
    pthread_join(tid[0], (void**)&sum0);
    pthread_join(tid[1], (void**)&sum1);
```

```
    printf("Sum of whole array = %i\n", *sum0 + *sum1);
    return 0;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ gcc thread.c -o a.out
sanjoy@SANJUVAI:~$ ./a.out
Sum of whole array = 55
```

10. ***Write a program to implement first-fit, best-fit and worst-fit allocation strategies.***

First-fit allocation strategies:

Program:

```
sanjoy@SANJUVAI:~$ gedit firstfit.c
// C implementation of First - Fit algorithm
#include<stdio.h>

// Function to allocate memory to blocks as per First fit algorithm
void firstFit(int blockSize[], int m, int processSize[], int n)
{
    int i, j;
    // Stores block id of the block allocated to a process
    int allocation[n];

    // Initially no block is assigned to any process
    for(i = 0; i < n; i++)
    {
            allocation[i] = -1;
    }
    for (i = 0; i < n; i++)        //here, n -> number of processes
    {
            for (j = 0; j < m; j++)        //here, m -> number of blocks
            {
                    if (blockSize[j] >= processSize[i])
                    {
                            // allocating block j to the ith process
                            allocation[i] = j;

                            // Reduce available memory in this block.
                            blockSize[j] -= processSize[i];

                            break; //go to the next process in the queue
```

```
                }
            }
        }

        printf("\nProcess No.\tProcess Size\tBlock no.\n");
        for (int i = 0; i < n; i++)
        {
            printf(" %i\t\t", i+1);
            printf("%i\t\t", processSize[i]);
            if (allocation[i] != -1)
                printf("%i", allocation[i] + 1);
            else
                printf("Not Allocated");
            printf("\n");
        }
}

int main()
{
    int m; //number of blocks in the memory
    int n; //number of processes in the input queue
    int blockSize[] = {100, 500, 200, 300, 600};
    int processSize[] = {212, 417, 112, 426};
    m = sizeof(blockSize) / sizeof(blockSize[0]);
    n = sizeof(processSize) / sizeof(processSize[0]);

    firstFit(blockSize, m, processSize, n);

    return 0 ;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ gcc firstfit.c -o a.out
sanjoy@SANJUVAI:~$ ./a.out

Process No.     Process Size    Block no.
 1              212             2
 2              417             5
 3              112             2
 4              426             Not Allocated
```

Best-fit allocation strategies:

Program:

```
sanjoy@SANJUVAI:~$ touch bestfit.cpp
// C++ implementation of Best - Fit algorithm
#include<iostream>
using namespace std;

// Method to allocate memory to blocks as per Best fit algorithm
void bestFit(int blockSize[], int m, int processSize[], int n)
{
    // Stores block id of the block allocated to a process
    int allocation[n];

    // Initially no block is assigned to any process
    for (int i = 0; i < n; i++)
            allocation[i] = -1;

    // pick each process and find suitable blocks according to its size ad
assign to it
    for (int i = 0; i < n; i++)
    {
            // Find the best fit block for current process
            int bestIdx = -1;
            for (int j = 0; j < m; j++)
            {
                    if (blockSize[j] >= processSize[i])
                    {
                            if (bestIdx == -1)
                                    bestIdx = j;
                            else if (blockSize[bestIdx] > blockSize[j])
                                    bestIdx = j;
                    }
            }

            // If we could find a block for current process
            if (bestIdx != -1)
            {
                    // allocate block j to p[i] process
                    allocation[i] = bestIdx;

                    // Reduce available memory in this block.
                    blockSize[bestIdx] -= processSize[i];
            }
    }
```

```
    cout << "\nProcess No.\tProcess Size\tBlock no.\n";
    for (int i = 0; i < n; i++)
    {
            cout << " " << i+1 << "\t\t" << processSize[i] << "\t\t";
            if (allocation[i] != -1)
                    cout << allocation[i] + 1;
            else
                    cout << "Not Allocated";
            cout << endl;
    }
}
int main()
{
    int blockSize[] = {100, 500, 200, 300, 600};
    int processSize[] = {212, 417, 112, 426};
    int m = sizeof(blockSize) / sizeof(blockSize[0]);
    int n = sizeof(processSize) / sizeof(processSize[0]);

    bestFit(blockSize, m, processSize, n);

    return 0 ;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ g++ bestfit.cpp
sanjoy@SANJUVAI:~$ ./a.out

Process No.     Process Size    Block no.
 1              212             4
 2              417             2
 3              112             3
 4              426             5
```

Worst-fit allocation strategies:

Program:

```
sanjoy@SANJUVAI:~$ gedit worstfit.cpp
// C++ implementation of worst - Fit algorithm
#include<bits/stdc++.h>
using namespace std;

// Function to allocate memory to blocks as per worst fit
// algorithm
 void worstFit(int blockSize[], int m, int processSize[],
```

```
    int n)
{
    // Stores block id of the block allocated to a
    // process
    int allocation[n];

    // Initially no block is assigned to any process
    memset(allocation, -1, sizeof(allocation));

    // pick each process and find suitable blocks
    // according to its size ad assign to it
    for (int i=0; i<n; i++)
    {
        // Find the best fit block for current process
        int wstIdx = -1;
        for (int j=0; j<m; j++)
        {
            if (blockSize[j] >= processSize[i])
            {
                if (wstIdx == -1)
                    wstIdx = j;
                else if (blockSize[wstIdx] < blockSize[j])
                    wstIdx = j;
            }
        }

        // If we could find a block for current process
        if (wstIdx != -1)
        {
            // allocate block j to p[i] process
            allocation[i] = wstIdx;

            // Reduce available memory in this block.
            blockSize[wstIdx] -= processSize[i];
        }
    }

    cout << "\nProcess No.\tProcess Size\tBlock no.\n";
    for (int i = 0; i < n; i++)
    {
        cout << " " << i+1 << "\t\t" << processSize[i] << "\t\t";
        if (allocation[i] != -1)
            cout << allocation[i] + 1;
```

```
            else
                cout << "Not Allocated";
            cout << endl;
    }
}

// Driver code
int main()
{
    int blockSize[] = {100, 500, 200, 300, 600};
    int processSize[] = {212, 417, 112, 426};
    int m = sizeof(blockSize)/sizeof(blockSize[0]);
    int n = sizeof(processSize)/sizeof(processSize[0]);
    worstFit(blockSize, m, processSize, n);
    return 0 ;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ g++ worstfit.cpp
sanjoy@SANJUVAI:~$ ./a.out

Process No.     Process Size    Block no.
 1              212             5
 2              417             2
 3              112             5
 4              426             Not Allocated
```

COMPUTER NETWORK LABORATORY MANUAL

(Course: BCAHMJ07P)

Assignments

1. Simulate Cyclic Redundancy Check (CRC) error detection algorithm for noisy channel.
2. Simulate and implement stop and wait protocol for noiseless channel.
3. Simulate and implement stop and wait protocol for noisy channel.
4. Simulate and implement go back n sliding window protocol.
5. Simulate and implement selective repeat sliding window protocol.
6. Simulate and implement Hamming Code error correction.
7. Simulate and implement distance vector routing algorithm.
8. Simulate and implement Dijkstra algorithm for shortest path routing.

1. Simulate Cyclic Redundancy Check (CRC) error detection algorithm for noisy channel.

Program:

```
#include <iostream>
#include <conio.h>
using namespace std;
int main()
{
   int i,j,k,l;

   //Get Frame
   int fs;
   cout<<"\n Enter Frame size: ";
   cin>>fs;

   int f[20];

   cout<<"\n Enter Frame:";
   for(i=0;i<fs;i++)
   {
      cin>>f[i];
   }

   //Get Generator
   int gs;
   cout<<"\n Enter Generator size: ";
   cin>>gs;

   int g[20];

   cout<<"\n Enter Generator:";
   for(i=0;i<gs;i++)
   {
      cin>>g[i];
   }

   cout<<"\n Sender Side:";
   cout<<"\n Frame: ";
   for(i=0;i<fs;i++)
   {
      cout<<f[i];
   }
```

```
cout<<"\n Generator :";
for(i=0;i<gs;i++)
{
   cout<<g[i];
}

//Append 0's
int rs=gs-1;
cout<<"\n Number of 0's to be appended: "<<rs;
for (i=fs;i<fs+rs;i++)
{
   f[i]=0;
}

int temp[20];
for(i=0;i<20;i++)
{
   temp[i]=f[i];
}

cout<<"\n Message after appending 0's :";
for(i=0; i<fs+rs;i++)
{
   cout<<temp[i];
}

//Division
for(i=0;i<fs;i++)
{
   j=0;
   k=i;
   //check whether it is divisible or not
   if (temp[k]>=g[j])
   {
      for(j=0,k=i;j<gs;j++,k++)
      {
         if((temp[k]==1 && g[j]==1) || (temp[k]==0 && g[j]==0))
         {
            temp[k]=0;
         }
         else
         {
            temp[k]=1;
```

```
            }
        }
    }
}

//CRC
int crc[15];
for(i=0,j=fs;i<rs;i++,j++)
{
    crc[i]=temp[j];
}

cout<<"\n CRC bits: ";
for(i=0;i<rs;i++)
{
    cout<<crc[i];
}

cout<<"\n Transmitted Frame: ";
int tf[15];
for(i=0;i<fs;i++)
{
    tf[i]=f[i];
}
for(i=fs,j=0;i<fs+rs;i++,j++)
{
    tf[i]=crc[j];
}
for(i=0;i<fs+rs;i++)
{
    cout<<tf[i];
}

cout<<"\n Receiver side : ";
cout<<"\n Received Frame: ";
for(i=0;i<fs+rs;i++)
{
    cout<<tf[i];
}

for(i=0;i<fs+rs;i++)
{
    temp[i]=tf[i];
```

```
}

//Division
for(i=0;i<fs+rs;i++)
{
   j=0;
   k=i;
   if (temp[k]>=g[j])
   {
      for(j=0,k=i;j<gs;j++,k++)
      {
         if((temp[k]==1 && g[j]==1) || (temp[k]==0 && g[j]==0))
         {
            temp[k]=0;
         }
         else
         {
            temp[k]=1;
         }
      }
   }
}

cout<<"\n Reaminder: ";
int rrem[15];
for (i=fs,j=0;i<fs+rs;i++,j++)
{
   rrem[j]= temp[i];
}
for(i=0;i<rs;i++)
{
   cout<<rrem[i];
}

int flag=0;
for(i=0;i<rs;i++)
{
   if(rrem[i]!=0)
   {
      flag=1;
   }
}
```

```
    if(flag==0)
    {
       cout<<"\n Since Remainder Is 0 Hence Message Transmitted From
Sender To Receiver Is Correct";
    }
    else
    {
       cout<<"\n Since Remainder Is Not 0 Hence Message Transmitted From
Sender To Receiver Contains Error";
    }
    getch();
}
```

Input and Output Section:

Enter Frame size: 8

Enter Frame:1 0 0 1 1 0 0 1

Enter Generator size: 4

Enter Generator:1 0 0 1

Sender Side:

Frame: 10011001

Generator :1001

Number of 0's to be appended: 3

Message after appending 0's :10011001000

CRC bits: 000

Transmitted Frame: 10011001000

Receiver side :

Received Frame: 10011001000

Remainder: 000

Since Remainder Is 0 Hence Message Transmitted From Sender To Receiver Is Correct

2. Simulate and implement stop and wait protocol for noiseless channel.

Program:

```
#include<iostream
>
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<dos.h>
using namespace
std; #define time 5
#define max_seq 1
#define tot_pack
5 int randn(int n)
{
return rand()%n + 1;
}
typedef struct
{
  int data;
}packet;
typedef
struct
{
   int kind;
   int seq;
   int ack;
   packet info;
}frame;
typedef enum{
frame_arrival,error,time_out}event_type; frame data1;
```

```
//creating prototype
void from_network_layer(packet
*); void to_physical_layer(frame
*); void to_network_layer(packet
*); void
from_physical_layer(frame*);
void sender();
void receiver();
void wait_for_event_sender(event_type *);
void wait_for_event_receiver(event_type
*);
//end
#define inc(k) if(k<max_seq)k++;else
k=0; int i=1;
char
turn; int
disc=0;
int
main()
{
  while(!disc)
  {  sender();
    // delay(400);
     receiver();
  }
  getchar();
}
  void sender()
{
   static int frame_to_send=0;
```

```
static frame s;
packet buffer;
event_type event;
```

```
    static int flag=0;       //first place
    if (flag==0)
    {
  from_network_layer(&buffer);
  s.info=buffer;
  s.seq=frame_to_send;
  cout<<"\nsender information \t"<<s.info.data<<"\n";
  cout<<"\nsequence no. \t"<<s.seq;
  turn='r';
  to_physical_layer(&s);
  flag=1;
    }
wait_for_event_sender(&event);
if(turn=='s')
{
    if(event==frame_arrival)
     {
     from_network_layer(&buffer);
     inc(frame_to_send);
     s.info=buffer;
     s.seq=frame_to_send;
     cout<<"\nsender information \t"<<s.info.data<<"\n";
     cout<<"\nsequence no. \t"<<s.seq<<"\n";
     getch();
     turn='r';
     to_physical_layer(&s);
     }
}
   }              //end of sender function
```

```
   void from_network_layer(packet *buffer)
{
    (*buffer).data=i;
    i++;
   }                    //end of from network layer function
   void to_physical_layer(frame *s)
{
    data1=*s;
   }        //end of to physical layer function
   void wait_for_event_sender(event_type
   *e)
{
     static int timer=0;
     if(turn=='s')
     {   timer++;
   //timer=0;
   return ;
  }
  else                    //event is frame arrival
   {
      timer=0;
      *e=frame_arrival;
   }
   }         //end of wait for event function
   void receiver()
{
    static int frame_expected=0;
    frame s,r;
    event_type event;
    wait_for_event_receiver(&event);
```

```
 if(turn=='r')
 {  if(event==frame_arrival)
{
     from_physical_layer(&r);
     if(r.seq==frame_expected)
{
   to_network_layer(&r.info);
   inc (frame_expected);
}
else
cout<<"\nReceiver :Acknowledgement resent \n";
getch();
turn='s';
to_physical_layer(&s);
  }
 }
}              //end of receiver function
void wait_for_event_receiver(event_type *e)
{
 if(turn=='r')
 {
*e=frame_arrival;
 }
}
void from_physical_layer(frame *buffer)
{
*buffer=data1;
}
void to_network_layer(packet *buffer)
```

```
{
    cout<<"\nReceiver : packet received \t"<< i-1;
    cout<<"\n Acknowledgement  sent \t";
    getch();
    if(i>tot_pack)
     { disc=1;
 cout<<"\ndiscontinue\n";
     }
   }    //end of network layer function
```

Input and Output Section:

```
   sender information  1
   sequence no.0
Receiver : packet received      1
 Acknowledgement  sent
   sender information  2
   sequence no.1


Receiver : packet received      2
 Acknowledgement  sent
   sender information  3
   sequence no.0
Receiver : packet received      3
 Acknowledgement  sent
```

sender information 4

sequence no.1

Receiver : packet received 4

Acknowledgement sent

sender information 5

sequence no.0

Receiver : packet received 5

Acknowledgement sent

Discontinue

3. Simulate and implement stop and wait protocol for noisy channel.

Program:

```
#include<iostream>
#include <time.h>
#include <cstdlib>
#include<ctime>
#include <unistd.h>
using namespace std;
class timer {
   private:
    unsigned long begTime;
   public:
    void start() {
     begTime = clock();
    }
  unsigned long elapsedTime() {
     return ((unsigned long) clock() - begTime) / CLOCKS_PER_SEC;
```

```
    }
   bool isTimeout(unsigned long seconds) {
      return seconds >= elapsedTime();
     }
};
int main()
{
 int frames[] = {1,2,3,4,5,6,7,8,9,10};
 unsigned long seconds = 5;
 srand(time(NULL));
 timer t;
 cout<<"Sender has to send frames : ";
 for(int i=0;i<10;i++)
    cout<<frames[i]<<" ";
 cout<<endl;
 int count = 0;
 bool delay = false;
 cout<<endl<<"Sender\t\t\t\tReceiver"<<endl;
 do
 {
    bool timeout = false;
    cout<<"Sending Frame : "<<frames[count];
    cout.flush();
    cout<<"\t\t";
    t.start();
    if(rand()%2)
    {
       int to = 24600 + rand()%(64000 - 24600)  + 1;
       for(int i=0;i<64000;i++)
```

```
        for(int j=0;j<to;j++) {}
    }
    if(t.elapsedTime() <= seconds)
    {
       cout<<"Received Frame : "<<frames[count]<<" ";
       if(delay)
       {
          cout<<"Duplicate";
          delay = false;
       }
       cout<<endl;
       count++;
    }
    else
    {
       cout<<"---"<<endl;
       cout<<"Timeout"<<endl;
       timeout = true;
    }
    t.start();
    if(rand()%2 || !timeout)
    {
       int to = 24600 + rand()%(64000 - 24600)  + 1;
       for(int i=0;i<64000;i++)
          for(int j=0;j<to;j++) {}
       if(t.elapsedTime() > seconds )
       {
          cout<<"Delayed Ack"<<endl;
          count--;
```

```
            delay = true;
        }
        else if(!timeout)
            cout<<"Acknowledgement : "<<frames[count]-1<<endl;
    }
  }while(count!=10);
  return 0;
 }
```

Input and Output Section:

Sender has to send frames : 1 2 3 4 5 6 7 8 9 10

Sender	Receiver
Sending Frame : 1	---
Timeout	
Sending Frame : 1	Received Frame : 1
Acknowledgement : 1	
Sending Frame : 2	Received Frame : 2
Acknowledgement : 2	
Sending Frame : 3	Received Frame : 3
Acknowledgement : 3	
Sending Frame : 4	---
Timeout	
Sending Frame : 4	Received Frame : 4
Acknowledgement : 4	
Sending Frame : 5	---
Timeout	

Sending Frame : 5 Received Frame : 5

Acknowledgement : 5

Sending Frame : 6 Received Frame : 6

Acknowledgement : 6

Sending Frame : 7 Received Frame : 7

Delayed Ack

Sending Frame : 7 ---

Timeout

Sending Frame : 7 Received Frame : 7 Duplicate

Delayed Ack

Sending Frame : 7 Received Frame : 7 Duplicate

Acknowledgement : 7

Sending Frame : 8 Received Frame : 8

Delayed Ack

Sending Frame : 8 ---

Timeout

Sending Frame : 8 Received Frame : 8 Duplicate

Delayed Ack

Sending Frame : 8 Received Frame : 8 Duplicate

Delayed Ack

Sending Frame : 8 ---

Timeout

Sending Frame : 8 Received Frame : 8 Duplicate

Delayed Ack

Sending Frame : 8 Received Frame : 8 Duplicate

Delayed Ack

Sending Frame : 8 ---

Timeout

Sending Frame : 8 ---

Timeout

Sending Frame : 8 Received Frame : 8 Duplicate

Acknowledgement : 8

Sending Frame : 9 ---

Timeout

Sending Frame : 9 Received Frame : 9

Acknowledgement : 9

Sending Frame : 10 Received Frame : 10

Delayed Ack

Sending Frame : 10 Received Frame : 10 Duplicate

Delayed Ack

Sending Frame : 10 Received Frame : 10 Duplicate

Acknowledgement : 0

4. Simulate and implement go back n sliding window protocol.

Program:

```
#include<iostream>
#include<ctime>
#include<cstdlib>
using namespace std;
int main()
{
 int nf,N;
 int no_tr=0;
 srand(time(NULL));
 cout<<"Enter the number of frames : ";
```

```
cin>>nf;
cout<<"Enter the Window Size : ";
cin>>N;
int i=1;
while(i<=nf)
{
   int x=0;
   for(int j=i;j<i+N && j<=nf;j++)
   {
      cout<<"Sent Frame "<<j<<endl;
      no_tr++;
   }
   for(int j=i;j<i+N && j<=nf;j++)
   {
      int flag = rand()%2;
      if(!flag)
         {
            cout<<"Acknowledgment for Frame "<<j<<endl;
            x++;
         }
      else
         {   cout<<"Frame "<<j<<" Not Received"<<endl;
            cout<<"Retransmitting Window"<<endl;
            break;
         }
   }
   cout<<endl;
   i+=x;
}
```

```
 cout<<"Total number of transmissions : "<<no_tr<<endl;
 return 0;
}
```

Input and Output Section:

Enter the number of frames : 9

Enter the Window Size : 3

Sent Frame 1

Sent Frame 2

Sent Frame 3

Frame 1 Not Received

Retransmitting Window

Sent Frame 1

Sent Frame 2

Sent Frame 3

Frame 1 Not Received

Retransmitting Window

Sent Frame 1

Sent Frame 2

Sent Frame 3

Frame 1 Not Received

Retransmitting Window

Sent Frame 1

Sent Frame 2

Sent Frame 3

Frame 1 Not Received

Retransmitting Window

Sent Frame 1

Sent Frame 2

Sent Frame 3

Acknowledgment for Frame 1

Frame 2 Not Received

Retransmitting Window

Sent Frame 2

Sent Frame 3

Sent Frame 4

Acknowledgment for Frame 2

Frame 3 Not Received

Retransmitting Window

Sent Frame 3

Sent Frame 4

Sent Frame 5

Acknowledgment for Frame 3

Acknowledgment for Frame 4

Acknowledgment for Frame 5

Sent Frame 6

Sent Frame 7

Sent Frame 8

Acknowledgment for Frame 6

Frame 7 Not Received

Retransmitting Window

Sent Frame 7

Sent Frame 8

Sent Frame 9

Acknowledgment for Frame 7

Acknowledgment for Frame 8

Frame 9 Not Received

Retransmitting Window

Sent Frame 9

Acknowledgment for Frame 9

Total number of transmissions : 28

5. Simulate and implement selective repeat sliding window protocol.

Program:

```
#include<iostream>
using namespace std;
#include<conio.h>
#include<stdlib.h>
#include<time.h>
#include<math.h>
#define TOT_FRAMES 500
#define FRAMES_SEND 10
class sel_repeat
{
private:
int fr_send_at_instance;
int arr[TOT_FRAMES];
int send[FRAMES_SEND];
int rcvd[FRAMES_SEND];

char rcvd_ack[FRAMES_SEND];
int sw;
int rw;       //tells expected frame
public:
```

```
void input();
void sender(int);
void receiver(int);
};
void sel_repeat::input()
{
int n;     //no. of bits for the frame
int m;    //no. of frames from n bits
int i;
cout<<"Enter the no. of bits for the sequence no. : ";
cin>>n;
m=pow(2,n);
int t=0;
fr_send_at_instance=(m/2);
for(i=0;i<TOT_FRAMES;i++)
{
arr[i]=t;
t=(t+1)%m;
}

for(i=0;i<fr_send_at_instance;i++)
{
send[i]=arr[i];
rcvd[i]=arr[i];
rcvd_ack[i]='n';
}
rw=sw=fr_send_at_instance;
sender(m);
}
```

```
void sel_repeat::sender(int m)
{
for(int i=0;i<fr_send_at_instance;i++)
{
if(rcvd_ack[i]=='n')
cout<<"SENDER : Frame "<<send[i]<<" is sent\n";
}
receiver(m);
}
void sel_repeat::receiver(int m)
{
time_t t;
int f;
int j;
int f1;
int a1;
char ch;
srand((unsigned)time(&t));
for(int i=0;i<fr_send_at_instance;i++)
{
if(rcvd_ack[i]=='n')
{
f=rand()%10;
//if f=5 frame is discarded for some reason
//else frame is correctly recieved
if(f!=5)
{
for(int j=0;j<fr_send_at_instance;j++)
if(rcvd[j]==send[i])
```

```
{
cout<<"reciever:Frame"<<rcvd[j]<<"recieved correctly\n";
rcvd[j]=arr[rw];
rw=(rw+1)%m;
break;
}
int j;
if(j==fr_send_at_instance)
cout<<"reciever:Duplicate frame"<<send[i]<<"discarded\n";
a1=rand()%5;
//if al==3 then ack is lost
//else recieved
if(a1==3)
{
cout<<"(acknowledgement "<<send[i]<<" lost)\n";
cout<<"(sender timeouts-->Resend the frame)\n";
rcvd_ack[i]='n';
}
else
{
cout<<"(acknowledgement "<<send[i]<<" recieved)\n";
rcvd_ack[i]='p';
}
}
else
{int ld=rand()%2;
//if =0 then frame damaged
//else frame lost
if(ld==0)
```

```
{
cout<<"RECEIVER : Frame "<<send[i]<<" is damaged\n";
cout<<"RECEIVER : Negative Acknowledgement "<<send[i]<<" sent\n";
}
else
{
cout<<"RECEIVER : Frame "<<send[i]<<" is lost\n";
cout<<"(SENDER TIMEOUTS-->RESEND THE FRAME)\n";
}
rcvd_ack[i]='n';
}
}
}
for(int j=0;j<fr_send_at_instance;j++)
{
if(rcvd_ack[j]=='n')
break;
}
int i=0;
for(int k=j;k<fr_send_at_instance;k++)
{
send[i]=send[k];

if(rcvd_ack[k]=='n')
rcvd_ack[i]='n';
else
rcvd_ack[i]='p';
i++;
```

```
}
if(i!=fr_send_at_instance)
{
for(int k=i;k<fr_send_at_instance;k++)
{
send[k]=arr[sw];
sw=(sw+1)%m;
rcvd_ack[k]='n';
}
}
cout<<"Want to continue";
cin>>ch;
cout<<"\n";
if(ch=='y')
sender(m);
else
exit(0);

}
int main()
{
sel_repeat sr;
sr.input();
}
```

Input and Output Section

Enter the no. of bits for the sequence no. : 4

SENDER : Frame 0 is sent

SENDER : Frame 1 is sent

SENDER : Frame 2 is sent

SENDER : Frame 3 is sent

SENDER : Frame 4 is sent

SENDER : Frame 5 is sent

SENDER : Frame 6 is sent

SENDER : Frame 7 is sent

reciever:Frame0recieved correctly

(acknowledgement 0 recieved)

reciever:Frame1recieved correctly

(acknowledgement 1 recieved)

reciever:Frame2recieved correctly

(acknowledgement 2 recieved)

reciever:Frame3recieved correctly

(acknowledgement 3 recieved)

reciever:Frame4recieved correctly

(acknowledgement 4 lost)

(sender timeouts-->Resend the frame)

reciever:Frame5recieved correctly

(acknowledgement 5 lost)

(sender timeouts-->Resend the frame)

reciever:Frame6recieved correctly

(acknowledgement 6 recieved)

reciever:Frame7recieved correctly

(acknowledgement 7 recieved)

Want to continue n

6. Simulate and implement Hamming Code error correction.

Program:

```
#include<iostream>
#include<cmath>
#include<string>
using namespace std;
class Hamming

{

 string message;
 int codeword[50],temp[50];
 int n,check;
 char parity;
public:
Hamming()
 {
    parity = 'E';
    message = "";
    n=check=0;
    for(int i=0;i<50;i++)
    {
       temp[i]=codeword[i]=0;
```

```
    }
}

void generate()
{
    do
    {
        cout<<"Enter the message in binary : ";
        cin>>message;
    }while(message.find_first_not_of("01") != string::npos);
    n=message.size();
    cout<<"Odd(O)/Even(E) Parity ? ";
    cin>>parity;
    for(unsigned int i=0;i<message.size();i++)
    {
        if(message[i] == '1')
            temp[i+1]=1;
        else
            temp[i+1]=0;
    }
    computeCode();
}
 void computeCode()
{
    check = findr();
    cout<<"Number of Check Bits : "<<check<<endl;
    cout<<"Number of Bits in Codeword : "<<n+check<<endl;
    for(int i=(n+check),j=n;i>0;i--)
    {
```

```
        if((i & (i - 1)) != 0)
            codeword[i] = temp[j--];
        else
            codeword[i] = setParity(i);
    }
    cout<<"Parity Bits - ";
    for(int i=0;i<check;i++)
     cout<<"P"<<pow(2,i)<<" : "<<codeword[(int)pow(2,i)]<<"\t";
    cout<<endl;
    cout<<"Codeword :"<<endl;
    for(int i=1;i<=(n+check);i++)
       cout<<codeword[i]<<" ";
    cout<<endl;
}
 int findr()
{
    for(int i=1;;i++)
    {
        if(n+i+1 <= pow(2,i))
            return i;
    } }
 int setParity(int x)
{
    bool flag = true;
    int bit;
    if(x == 1)
    {
        bit = codeword[x+2];
        for(int j=x+3;j<=(n+check);j++)
```

```
        {
            if(j%2)
            {
                bit ^= codeword[j];
            }
        }
    }
    else
    {
        bit = codeword[x+1];
        for(int i=x;i<=(n+check);i++)
        {
            if(flag)
            {
                if(i==x || i==x+1)
                    bit = codeword[x+1];
                else
                    bit ^= codeword[i];
            }
            if((i+1)%x == 0)
                flag = !flag;
        }
    }
    if(parity == 'O' || parity == 'o')
        return !bit;
    else
        return bit;
}
    void correct()
```

```
{
   do
   {
      cout<<"Enter the received codeword : ";
      cin>>message;
   }while(message.find_first_not_of("01") != string::npos);
   for(unsigned int i=0;i<message.size();i++)
   {
      if(message[i] == '1')
         codeword[i+1]=1;
      else
         codeword[i+1]=0;
   }
   detect();
}
void detect()
{
   int position = 0;
   cout<<"Parity Bits - ";
   for(int i=0;i<check;i++)
   {
      bool flag = true;
      int x = pow(2,i);
      int bit = codeword[x];
      if(x == 1)
      {
         for(int j=x+1;j<=(n+check);j++)
         {
            if(j%2)
```

```
            {
                bit ^= codeword[j];
            }
        }
    }
    else
    {
        for(int k=x+1;k<=(n+check);k++)
        {
            if(flag)
            {
                bit ^= codeword[k];
            }
            if((k+1)%x == 0)
                flag = !flag;
        }
    }
    cout<<"P"<<x<<": "<<bit<<"\t";
    if((parity=='E' || parity == 'e') && bit==1)
        position += x;
    if((parity=='O' || parity == 'o') && bit==0)
        position += x;
}
cout<<endl<<"Received Codeword :"<<endl;
for(int i=1;i<=(n+check);i++)
    cout<<codeword[i]<<" ";
cout<<endl;
if(position != 0)
{
```

```
            cout<<"Error at bit : "<<position<<endl;
            codeword[position] = !codeword[position];
            cout<<"Corrected Codeword : "<<endl;
            for(int i=1;i<=(n+check);i++)
                cout<<codeword[i]<<" ";
            cout<<endl;
        }
        else
            cout<<"No Error in Received code."<<endl;
        cout<<"Received Message is : ";
        for(int i=1;i<=(n+check);i++)
            if((i & (i - 1)) != 0)
                cout<<codeword[i]<<" ";
        cout<<endl;
    }
};
int main()
{
 char choice;
 do
 {
    Hamming a;
    cout<<"At Sender's side : "<<endl;
    a.generate();
    cout<<endl<<"At Receiver's Side : "<<endl;
    a.correct();
    cout<<endl<<"Enter another code ? (Y/N) : ";
    cin>>choice;
    cout<<endl;
```

```
 }while(choice == 'y' || choice == 'Y');
 return 0;
}
```

Input and Output Section:

At Sender's side :

Enter the message in binary : 1001101

Odd(O)/Even(E) Parity ? E

Number of Check Bits : 4

Number of Bits in Codeword : 11

Parity Bits - P1 : 0 P2 : 1 P4 : 1 P8 : 0

Codeword :

0 1 1 1 0 0 1 0 1 0 1

At Receiver's Side :

Enter the received codeword : 01110010101

Parity Bits - P1: 0 P2: 0 P4: 0 P8: 0

Received Codeword :

0 1 1 1 0 0 1 0 1 0 1

No Error in Received code.

Received Message is : 1 0 0 1 1 0 1

Enter another code ? (Y/N) : N

7. Simulate and implement distance vector routing algorithm.

Program:

```
#include<stdio.h>
#include<iostream>
using namespace std;
struct node
{
   unsigned dist[6];
   unsigned from[6];
}DVR[10];
int main()
{
   cout<<"\n\n PROGRAM TO IMPLEMENT DISTANCE VECTOR
ROUTING ALGORITHM ";
   int costmat[6][6];
   int nodes, i, j, k;
   cout<<"\n\n Enter the number of nodes : ";
   cin>>nodes; //Enter the nodes
   cout<<"\n Enter the cost matrix : \n" ;
   for(i = 0; i < nodes; i++)
    {
      for(j = 0; j < nodes; j++)
      {
         cin>>costmat[i][j];
         costmat[i][i] = 0;
         DVR[i].dist[j] = costmat[i][j]; //initialise the distance equal to cost
matrix
         DVR[i].from[j] = j;
```

```
        }
    }
        for(i = 0; i < nodes; i++) //We choose arbitary vertex k and we
calculate the
        //direct distance from the node i to k using the cost matrix and add
the distance from k to node j
        for(j = i+1; j < nodes; j++)
        for(k = 0; k < nodes; k++)
           if(DVR[i].dist[j] > costmat[i][k] + DVR[k].dist[j])
           {   //We calculate the minimum distance
              DVR[i].dist[j] = DVR[i].dist[k] + DVR[k].dist[j];
              DVR[j].dist[i] = DVR[i].dist[j];
              DVR[i].from[j] = k;
              DVR[j].from[i] = k;

           }
      for(i = 0; i < nodes; i++)
      {
        cout<<"\n\n For router: "<<i+1;
        for(j = 0; j < nodes; j++)
           cout<<"\t\n node "<<j+1<<" via "<<DVR[i].from[j]+1<<"
Distance "<<DVR[i].dist[j];
      }
   cout<<" \n\n ";
   return 0;
}
```

Input and Output Section:

PROGRAM TO IMPLEMENT DISTANCE VECTOR ROUTING ALGORITHM

Enter the number of nodes : 3

Enter the cost matrix :

0

2

7

2

0

1

7

1

0

For router: 1

node 1 via 1 Distance 0

node 2 via 2 Distance 2

node 3 via 2 Distance 3

For router: 2

node 1 via 1 Distance 2

node 2 via 2 Distance 0

node 3 via 3 Distance 1

For router: 3

node 1 via 2 Distance 3

node 2 via 2 Distance 1

node 3 via 3 Distance 0

8. Simulate and implement Dijkstra algorithm for shortest path routing.

Program:

```
#include<iostream>
#include<conio.h>
#include<stdio.h>
using namespace std;
int shortest(int ,int);
int cost[10][10],dist[20],i,j,n,k,m,S[20],v,totcost,path[20],p;
main()
{
int c;
cout <<"enter no of vertices";
cin >> n;
cout <<"enter no of edges";
```

```
cin >>m;
cout <<"\nenter\nEDGE Cost\n";
for(k=1;k<=m;k++)
{
cin >> i >> j >>c;
cost[i][j]=c;
}
for(i=1;i<=n;i++)
for(j=1;j<=n;j++)
if(cost[i][j]==0)
cost[i][j]=31999;
cout <<"enter initial vertex";
cin >>v;
cout << v<<"\n";
shortest(v,n);
 }

int shortest(int v,int n)
{
int min;
for(i=1;i<=n;i++)
{
S[i]=0;
dist[i]=cost[v][i];
}
path[++p]=v;
S[v]=1;
dist[v]=0;
for(i=2;i<=n-1;i++)
```

```
{
k=-1;
min=31999;
for(j=1;j<=n;j++)
{
if(dist[j]<min && S[j]!=1)
{
min=dist[j];
k=j;
}
}
if(cost[v][k]<=dist[k])
p=1;
path[++p]=k;
for(j=1;j<=p;j++)
cout<<path[j];
cout <<"\n";
//cout <<k;
S[k]=1;
for(j=1;j<=n;j++)
if(cost[k][j]!=31999 && dist[j]>=dist[k]+cost[k][j] && S[j]!=1)
 dist[j]=dist[k]+cost[k][j];
}
}
```

Input and Output Section:

enter no of vertices5

enter no of edges7

enter

EDGE Cost

1

2

10

1

3

2

1

5

100

2

4

3

3

25

5

4

5

5

4

3

15

enter initial vertex1 1

13

12

124