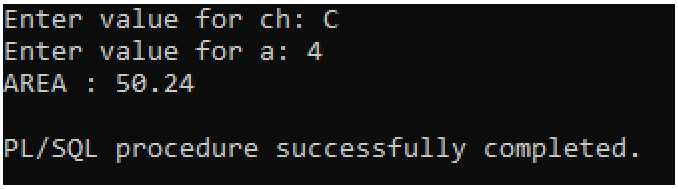
GROUP – A

Question 1

A.

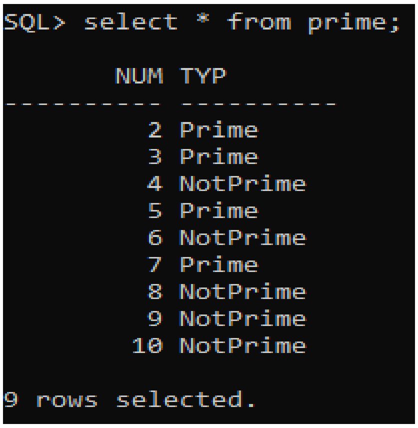
```
Create or replace function FAREA(x in number, c in char )
return number
IS
ar number;
BEGIN
if (c='C' or c = 'c') then
ar:=3.14*x*x;
else
ar:=x*x;
END IF;
RETURN ar;
END FAREA;
/

DECLARE
a number;
ch char;
arr number;
Invaid_input EXCEPTION;
BEGIN
ch:= '&ch';
a:=&a;
if((ch ='C' or ch = 'c') or (ch ='S' or ch = 's') ) then
arr := FAREA(a,ch);
DBMS_OUTPUT.PUT_LINE('AREA : ' || ROUND (arr,2));
else
RAISE Invaid_input;
end if;
EXCEPTION
when Invaid_input then
DBMS_OUTPUT.PUT_LINE('ERROR :: WRONG INPUT .. pls enter only C or S');
when NO_DATA_FOUND then
DBMS_OUTPUT.PUT_LINE('ERROR :: INVALID .');
end;
/
```



```
Enter value for ch: C
Enter value for a: 4
AREA : 50.24

PL/SQL procedure successfully completed.
```

B.

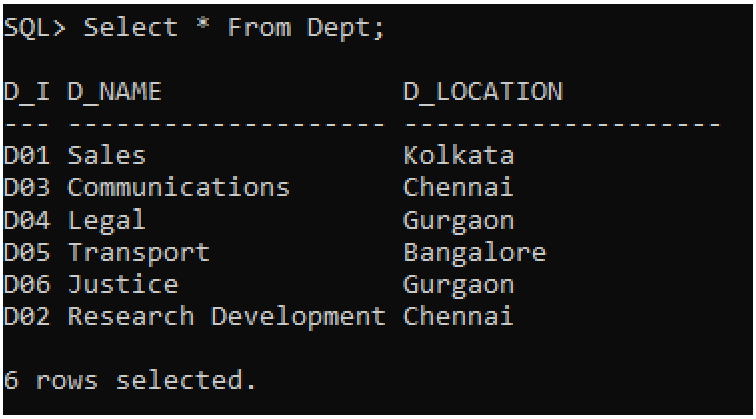
```
create table prime
(
num number,
typ varchar2(10)
);

DECLARE
num number(2);
n number(2);
m number(2);
c number(20);
BEGIN
dbms_output.put_line('limit Entered.');
num:=&num;
for n IN 2..num
loop
c:=0;
for m IN 1.. n
loop
if mod(n, m)=0 then
c:=c+1;
end if;
end loop;
if c<=2 then
insert into prime values (n,'Prime');
else
insert into prime values (n,'NotPrime');
end if;
end loop;
END;
```



```
SQL> select * from prime;

       NUM TYP
---------- ----------
         2 Prime
         3 Prime
         4 NotPrime
         5 Prime
         6 NotPrime
         7 Prime
         8 NotPrime
         9 NotPrime
        10 NotPrime

9 rows selected.
```

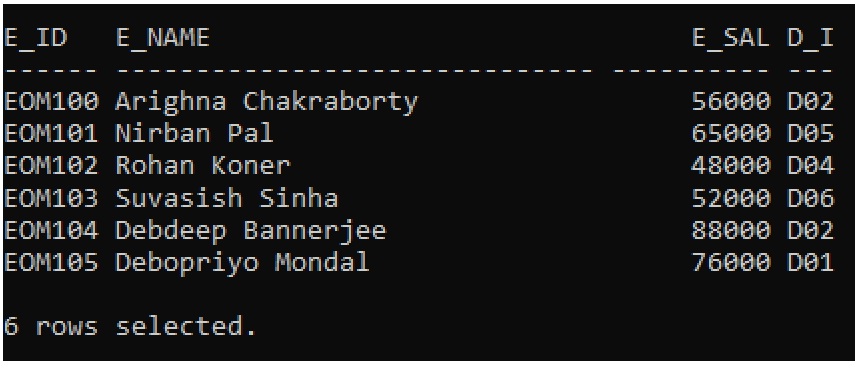
Question 2

```
CREATE TABLE Dept(
d_id VARCHAR2(3) PRIMARY KEY,
d_name VARCHAR2(20) NOT NULL,
d_location VARCHAR2(20) CHECK (d_location = 'Kolkata' OR d_location = 'Bangalore' OR
d_location = 'Chennai' OR d_location = 'Gurgaon')
);

INSERT INTO Dept VALUES('D01', 'Sales', 'Kolkata');
INSERT INTO Dept VALUES('D02', 'Research Development', 'Chennai');
INSERT INTO Dept VALUES('D03', 'Communications', 'Chennai');
INSERT INTO Dept VALUES('D04', 'Legal', 'Gurgaon');
INSERT INTO Dept VALUES('D05', 'Transport', 'Bangalore');
INSERT INTO Dept VALUES('D06', 'Justice', 'Gurgaon');

SELECT * FROM Dept;
```



```
SQL> Select * From Dept;

D_I D_NAME               D_LOCATION
--- -------------------- --------------------
D01 Sales                Kolkata
D03 Communications       Chennai
D04 Legal                Gurgaon
D05 Transport            Bangalore
D06 Justice              Gurgaon
D02 Research Development Chennai

6 rows selected.
```

```
CREATE TABLE Emp(
e_id VARCHAR2(6) CHECK (e_id like 'EOM%'),
e_name VARCHAR2(30) NOT NULL,
e_sal INT CHECK (e_sal BETWEEN 15000 AND 150000),
d_id VARCHAR(3),
PRIMARY KEY(e_id),
FOREIGN KEY(d_id) REFERENCES Dept(d_id)
);

INSERT INTO Emp VALUES('EOM100', 'Arighna Chakraborty', 56000, 'D02');
INSERT INTO Emp VALUES('EOM101', 'Nirban Pal', 65000, 'D05');
INSERT INTO Emp VALUES('EOM102', 'Rohan Koner', 48000, 'D04');
INSERT INTO Emp VALUES('EOM103', 'Suvasish Sinha', 52000, 'D06');
INSERT INTO Emp VALUES('EOM104', 'Debdeep Bannerjee', 88000, 'D02');
INSERT INTO Emp VALUES('EOM105', 'Debopriyo Mondal', 76000, 'D01');
```



```
E_ID   E_NAME                              E_SAL D_I
------ ------------------------------ ---------- ---
EOM100 Arighna Chakraborty                 56000 D02
EOM101 Nirban Pal                          65000 D05
EOM102 Rohan Koner                         48000 D04
EOM103 Suvasish Sinha                      52000 D06
EOM104 Debdeep Bannerjee                   88000 D02
EOM105 Debopriyo Mondal                    76000 D01

6 rows selected.
```

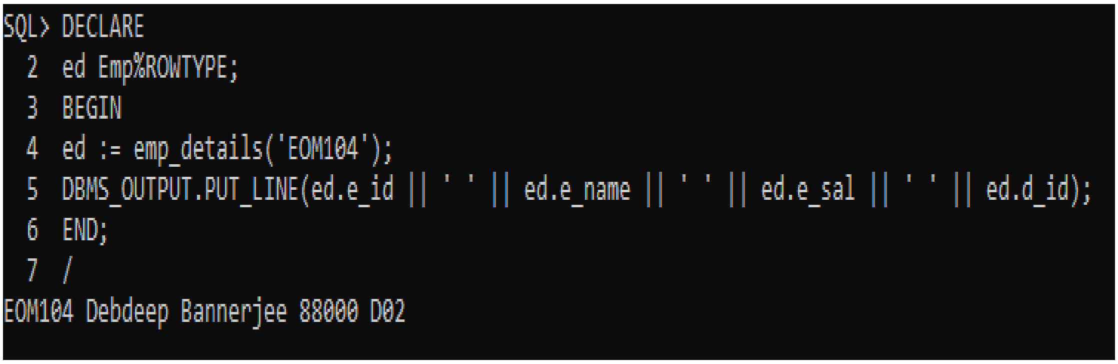
A.

```
DECLARE
CURSOR cur_sal IS
SELECT Emp.e_id, Emp.e_sal, Dept.d_location FROM Emp
INNER JOIN Dept ON Emp.d_id = Dept.d_id;
v_id VARCHAR2(6);
v_sal INT;
v_sal_buffer INT;
v_loc VARCHAR2(20);
BEGIN
OPEN cur_sal;
LOOP
FETCH cur_sal INTO v_id, v_sal, v_loc;
EXIT WHEN cur_sal%NOTFOUND;
v_sal_buffer := v_sal;
```

```
IF v_loc = 'Gurgaon' THEN
v_sal := v_sal - (0.05 * v_sal);
ELSE IF v_loc = 'Chennai' THEN
v_sal := v_sal + (0.15 * v_sal);
ELSE
v_sal := vsal;
END IF;
IF v_sal <= 15000 OR v_saal >= 150000 THEN
DBMS_OUTPUT.PUT_LINE('Salary exceeds validation limit. Fees restored');
v_sal := v_sal_buffer;
END IF;
UPDATE Emp
SET e_sal = v_sal
WHERE e_id = v_id;
END LOOP;
CLOSE cur_sal;
END;
/
```

B.

```
CREATE OR REPLACE FUNCTION emp_details(v_e_id IN VARCHAR2) RETURN
Emp%ROWTYPE
AS
e_record Emp%ROWTYPE;
BEGIN
SELECT * INTO e_record FROM Emp WHERE e_id = v_e_id;
RETURN e_record;
EXCEPTION
WHEN NO_DATA_FOUND THEN
DBMS_OUTPUT.PUT_LINE('No Such Employee Exists...');
END;
/
```



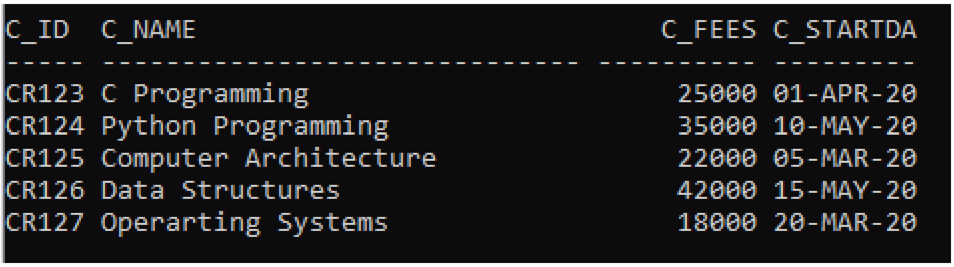
```
SQL> DECLARE
  2  ed Emp%ROWTYPE;
  3  BEGIN
  4  ed := emp_details('EOM104');
  5  DBMS_OUTPUT.PUT_LINE(ed.e_id || ' ' || ed.e_name || ' ' || ed.e_sal || ' ' || ed.d_id);
  6  END;
  7  /
EOM104 Debdeep Bannerjee 88000 D02
```

Question 3

```
CREATE TABLE Course(
c_id VARCHAR2(5) PRIMARY KEY CHECK (c_id like 'CR%'),
c_name VARCHAR2(30) NOT NULL,
c_fees INT CHECK (c_fees BETWEEN 5000 AND 50000),
c_startdate DATE
);
```

```
INSERT INTO Course VALUES('CR123', 'C Programming', 25000, TO_DATE('01-04-2020', 'DD/MM/YYYY'));
INSERT INTO Course VALUES('CR124', 'Python Programming', 35000, TO_DATE('10-05-2020', 'DD/MM/YYYY'));
INSERT INTO Course VALUES('CR125', 'Computer Architecture', 22000, TO_DATE('05-03-2020', 'DD/MM/YYYY'));
INSERT INTO Course VALUES('CR126', 'Data Structures', 42000, TO_DATE('15-05-2020', 'DD/MM/YYYY'));
INSERT INTO Course VALUES('CR127', 'Operarting Systems', 18000, TO_DATE('20-03-2020', 'DD/MM/YYYY'));
```

```
SELECT * FROM Course;
```

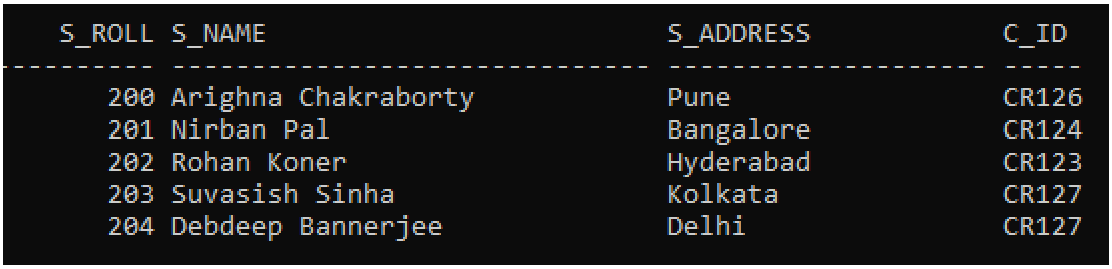


```
C_ID  C_NAME                              C_FEES C_STARTDA
----- ------------------------------ ---------- ---------
CR123 C Programming                       25000 01-APR-20
CR124 Python Programming                  35000 10-MAY-20
CR125 Computer Architecture               22000 05-MAR-20
CR126 Data Structures                     42000 15-MAY-20
CR127 Operarting Systems                  18000 20-MAR-20
```

```
CREATE TABLE Student(
s_roll NUMBER(3) PRIMARY KEY,
s_name VARCHAR2(30) NOT NULL,
s_address VARCHAR2(20),
c_id VARCHAR2(5) CHECK (c_id like 'CR%'),
FOREIGN KEY(c_id) REFERENCES Course(c_id)
);
```

```
INSERT INTO Student VALUES(200, 'Arighna Chakraborty', 'Pune', 'CR126');
INSERT INTO Student VALUES(201, 'Nirban Pal', 'Bangalore', 'CR124');
INSERT INTO Student VALUES(202, 'Rohan Koner', 'Hyderabad', 'CR123');
INSERT INTO Student VALUES(203, 'Suvasish Sinha', 'Kolkata', 'CR127');
INSERT INTO Student VALUES(204, 'Debdeep Bannerjee', 'Delhi', 'CR127');
```

SELECT * FROM Student;



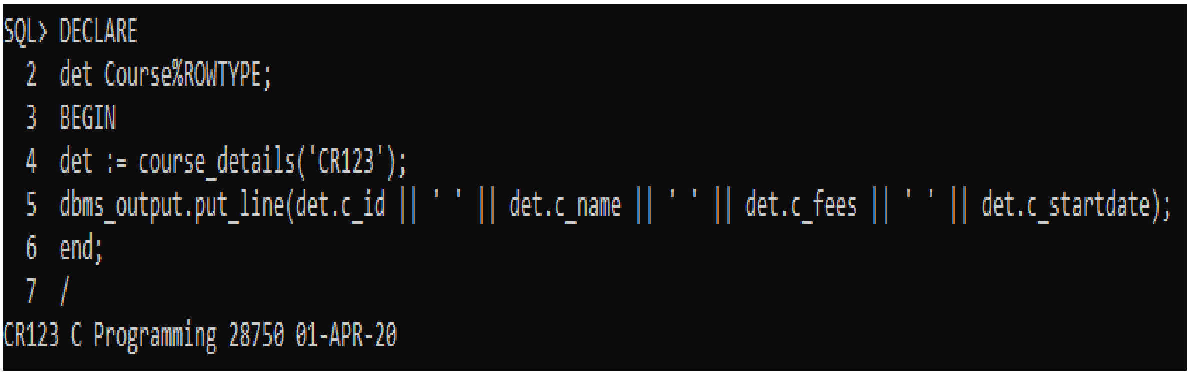
```
    S_ROLL S_NAME                         S_ADDRESS            C_ID
---------- ------------------------------ -------------------- -----
       200 Arighna Chakraborty            Pune                 CR126
       201 Nirban Pal                     Bangalore            CR124
       202 Rohan Koner                    Hyderabad            CR123
       203 Suvasish Sinha                 Kolkata              CR127
       204 Debdeep Bannerjee              Delhi                CR127
```

A.

```
DECLARE
CURSOR cur_fees IS
SELECT c_name, c_fees from Course;
v_name VARCHAR2(30);
v_fees INT;
v_fees_buffer INT;
BEGIN
OPEN cur_fees;
LOOP
FETCH cur_fees INTO v_name, v_fees;
EXIT WHEN cur_fees%NOTFOUND;
v_fees_buffer := v_fees;
IF v_name = 'Python Programming' THEN
v_fees := v_fees + (0.1 * v_fees);
ELSE
v_fees := v_fees + (0.15 * v_fees);
END IF;
IF v_fees <= 5000 OR v_fees >= 50000 THEN
DBMS_OUTPUT.PUT_LINE('Fees exceeds validation limit. Fees restored');
v_fees := v_fees_buffer;
END IF;
UPDATE Course
SET c_fees = v_fees
WHERE c_name = v_name;
END LOOP;
CLOSE cur_fees;
END;
/
```

B.

```
CREATE OR REPLACE FUNCTION course_details(v_c_id IN VARCHAR2) RETURN
Course%ROWTYPE
AS
course_record Course%ROWTYPE;
BEGIN
SELECT * INTO course_record FROM Course WHERE c_id = v_c_id;
RETURN course_record;
EXCEPTION
WHEN NO_DATA_FOUND THEN
DBMS_OUTPUT.PUT_LINE('No Such Course Available...');
END;
/
```



```
SQL> DECLARE
  det Course%ROWTYPE;
  BEGIN
  det := course_details('CR123');
  dbms_output.put_line(det.c_id || ' ' || det.c_name || ' ' || det.c_fees || ' ' || det.c_startdate);
  end;
  /
CR123 C Programming 28750 01-APR-20
```

GROUP - B

5.a)

```
#include<stdio.h>
#include<omp.h>
#include<sys/types.h>
int main(void){
                int i, n , m, R;
                printf("\n Enter any value: ");
                scanf("%d" , &n);
                omp_set_dynamic(0);
                m= omp_get_num_procs();
                omp_set_num_threads(m);
                #pragma omp parallel for reduction (+: R)
                for ( i = 1 ; i<= n ; i++) {
                        R += i*i;
```

```
		}
		printf("n Sum of the given series till %d is:  %d\n", n, R);
		return 0 ;
}
```

5.b)

```
#include<stdio.h>
#include<unistd.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<sys/wait.h>
#include<fcntl.h>
#define size 1024
int main()
{
		int fd,n;
		char buff[size];
		fd=open("sample.txt",O_RDONLY,0777);
		if(fd==-1)
		{
			printf("\n File not found.");
		}
		else
		{
			n=read(fd,buff,size);
			printf("The content of the file is: \n");
			write(0,buff,n);
		}
		close(fd);
}
```

6.a)

```
#include<stdio.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<fcntl.h>
#include<unistd.h>
#include<stdlib.h>
#include<string.h>
void main()
{
		int fd;
		fd=fork();
		char *args[] = {"./date",NULL};
		if(fd==0)
			execv(args[0], args);

}
#include<stdio.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<fcntl.h>
#include<unistd.h>
#include<stdlib.h>
#include<string.h>
int main()
{
		system("date");
}
```

6.b

```
#include<stdio.h>
#include<omp.h>
void main()
{
		int i,j;
		int a[10][10],b[10][10],c[10][10],sum1;
		float d[10],avg;
		int n,sum;
		printf("\nEnter the size of the square matrix:");
		scanf("%d",&n);
		printf("\nEnter values of matrix A:");
		for(i=0;i<n;i++)
			for(j=0;j<n;j++)
			{
				scanf("%d",&a[i][j]);
			}
		printf("\nEnter values of matrix B:");
		for(i=0;i<n;i++)
			for(j=0;j<n;j++)
			{
				scanf("%d",&b[i][j]);
			}
		printf("\nThe values of the matrix A are:\n");
```

```
        for(i=0;i<n;i++)
        {
            for(j=0;j<n;j++)
            {
                printf("\t%d\t",a[i][j]);
            }
            printf("\n");
        }
        printf("\nThe values of the matrix B are:\n");
        for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
        {
            printf("\t%d\t",b[i][j]);
        }
        printf("\n");
    }

    for(i=0;i<n;i++)
    {
        #pragma omp parallel for default(none), private(j), shared(a,b,c,n,sum,d,i)
        for(j=0;j<n;j++)
        {
        sum=a[i][j]+b[i][j];
        c[i][j]=sum;

        printf("\nIn thread=%d c[%d][%d]\n",omp_get_thread_num(),i,j);
        }
    }
    printf("\nsum of the matrix\n");
    for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
        {
            printf("\t%d\t",c[i][j]);
        }
        printf("\n");
    }

    sum=0;
            for(i=0;i<n;i++)
            {

                for(j=0;j<n;j++)
                {
                    sum=sum+c[i][j];

                    avg=(float)sum/n;
                }
            printf("\nsum of row %d is %d",i,sum);
            printf("\n average of row %d is %f",i,avg);
            sum=0;
            printf("\n");
        }
        for(j=0;j<n;j++)
        {
            for(i=0;i<n;i++)
            {
                sum=sum+c[i][j];

                avg=(float)sum/n;
            }
            printf("\nsum of column %d is %d",i,sum);
            printf("\n average of column %d is %f",i,avg);
            sum=0;
        printf("\n");
    }
}
```

8.a)

```
#include<stdio.h>
#include<unistd.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<sys/wait.h>
#include<fcntl.h>
#define size 1024
int main()
{
int fd,n,cid,status;
        char buff[size];
cid=fork();
if(cid==0)
{
    printf("Enter the text: \n");
    n=read(0,buff,size);
    fd=open("sample.txt",O_CREAT|O_WRONLY,0777);
    if(fd==-1)
    {
        printf("\n File is not created successfully");
    }
    else
    {
        write(fd,buff,n);
    }
    close(fd);
    }
    else
    {
        wait(&status);
        fd=open("sample.txt",O_CREAT|O_RDONLY,0777);
        if(fd==-1)
        {
            printf("\n File not found.");
        }
        else
            {
                printf("The content of the file is: \n");
                lseek(fd,0,SEEK_SET);
                n=read(fd,buff,size);
                write(0,buff,n);
            }
            close(fd);
        }
}
```

8.b)

```
#include<stdio.h>
#include<omp.h>
void main()
{
    int i,j;
    int a[10][10],b[10][10],c[10][10],sum1;
    float d[10],avg;
    int n,sum;
```

```
printf("\nEnter the size of the square matrix:");
scanf("%d",&n);
printf("\nEnter values of matrix A:");
for(i=0;i<n;i++)
        for(j=0;j<n;j++)
        {
                scanf("%d",&a[i][j]);
        }
printf("\nEnter values of matrix B:");
for(i=0;i<n;i++)
        for(j=0;j<n;j++)
        {
                scanf("%d",&b[i][j]);
        }
printf("\nThe values of the matrix A are:\n");
for(i=0;i<n;i++)
{
        for(j=0;j<n;j++)
        {
                printf("\t%d\t",a[i][j]);
        }
        printf("\n");
}
printf("\nThe values of the matrix B are:\n");
for(i=0;i<n;i++)
{
        for(j=0;j<n;j++)
        {
                printf("\t%d\t",b[i][j]);
        }
        printf("\n");
}
for(i=0;i<n;i++)
{
        #pragma omp parallel for default(none), private(j), shared(a,b,c,n,sum,d,i)
        for(j=0;j<n;j++)
        {
        sum=a[i][j]+b[i][j];
        c[i][j]=sum;
        printf("\nIn thread=%d c[%d][%d]\n",omp_get_thread_num(),i,j);
        }
}
```

```
    printf("\nsum of the matrix\n");
    for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
        {
            printf("\t%d\t",c[i][j]);
        }
        printf("\n");
    }
    sum=0;
    for(i=0;i<n;i++)
    {
            for(j=0;j<n;j++)
            {
                sum=sum+c[i][j];
                avg=(float)sum/n;
            }
            printf("\nsum of row %d is %d",i,sum);
            printf("\n average of row %d is %f",i,avg);
            sum=0;
        printf("\n");
    }
    for(j=0;j<n;j++)
    {
            for(i=0;i<n;i++)
            {
                sum=sum+c[i][j];
                avg=(float)sum/n;
            }
            printf("\nsum of column %d is %d",i,sum);
            printf("\n average of column %d is %f",i,avg);
            sum=0;
            printf("\n");
    }
}
```