```
-- GR - A
-- 1. b

--
 Write a PL/SQL code to store the first n positive integers along with their cubes in an
already created
--
 table. 'n' should be taken as an input from the user. The program will also display the
output of the
-- table

--table creation

create table cubes_of_numbers(input_number number,cube_of_number number);

--plsql code goes here

set serveroutput on
set verify off

declare
        limit_var cubes_of_numbers.input_number%type;
        c_o_n cubes_of_numbers.cube_of_number%type;
        loop_var cubes_of_numbers.input_number%type;
        content_of_table cubes_of_numbers%rowtype;
        cursor cur_cubes_of_numbers
        is
        select * from cubes_of_numbers;
begin
        limit_var := &limit_of_entry;
        for loop_var in 1..limit_var
        loop
            c_o_n := power(loop_var, 3);
            insert into cubes_of_numbers values(loop_var, c_o_n);
        end loop;
        open cur_cubes_of_numbers;
        dbms_output.put_line('Content of the table goes here ----');
        dbms_output.put_line('-----------------------------------');
        dbms_output.put_line('input_number cube_of_number');
        dbms_output.put_line('-----------------------------------');
        loop
            fetch cur_cubes_of_numbers into content_of_table;
            if cur_cubes_of_numbers%notfound then
                exit;
            else
                dbms_output.put_line(content_of_table.input_number || '               ' ||
 content_of_table.cube_of_number);
            end if;
        end loop;
        close cur_cubes_of_numbers;
```

```
end;
/

-- 1.a

--
 Write a Function FAREA to calculate the Area of a Square or Area of a Circle. It accepts
 a number
--
 and a character parameter. The character parameter is either ‘C’ to compute area of a Ci
rcle or ‘S’ to
-- compute the Area of a Square. Raise an Exception in case of invalid input.

declare
    input_val number;
    input_choice varchar2(1);
    area number;
    invalid_choice exception;
    function FAREA(val in number, choice in varchar2)
    return number
    is
        area number;
        pi constant number := 3.14;
        circle_choice varchar2(1) := 'C';
        square_choice varchar2(1) := 'S';
    begin
        if choice = circle_choice then
            area := pi * val * val;
        elsif choice = square_choice then
            area := val * val;
        else
            raise invalid_choice;
        end if;
        return area;
        exception
            when invalid_choice then
                dbms_output.put_line('You have chosen wrong input. Do check and choose be
tween "S" and "C"');
                return -1;
    end;
begin
    input_val := &input_number;
    input_choice := '&input_choice';
    area := FAREA(input_val, input_choice);
    if area != -1 then
        dbms_output.put_line('Area : ' || area);
    end if;
end;
/
```

```
25                  return -1;
26      end;
27  begin
28      input_val := &input_number;
29      input_choice := '&input_choice';
30      area := FAREA(input_val, input_choice);
31      if area != -1 then
32          dbms_output.put_line('Area : ' || area);
33      end if;
34  end;
35  /
Enter value for input_number: 4
Enter value for input_choice: D
You have chosen wrong input. Do check and choose between "S" and "C"

PL/SQL procedure successfully completed.

SQL>
```

```
  29      input_choice := '&input_choice';
  30      area := FAREA(input_val, input_choice);
  31      if area != -1 then
  32          dbms_output.put_line('Area : ' || area);
  33      end if;
  34  end;
  35  /
Enter value for input_number: 4
Enter value for input_choice: C
Area : 50.24

PL/SQL procedure successfully completed.
```

```
28      input_val := &input_number;
29      input_choice := '&input_choice';
30      area := FAREA(input_val, input_choice);
31      if area != -1 then
32          dbms_output.put_line('Area : ' || area);
33      end if;
34  end;
35  /
Enter value for input_number: 4
Enter value for input_choice: S
Area : 16

PL/SQL procedure successfully completed.

SQL>
```

```
  26                dbms_output.put_line(content_of_table.input_number
of_number);
  27              end if;
  28          end loop;
  29          close cur_cubes_of_numbers;
  30  end;
  31  /
Enter value for limit_of_entry: 9
Content of the table goes here ----
-----------------------------------
input_number cube_of_number
-----------------------------------
1              1
2              8
3              27
4              64
5              125
6              216
7              343
8              512
9              729

PL/SQL procedure successfully completed.

SQL>
```

```
SQL> desc cubes_of_numbers;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 INPUT_NUMBER                                       NUMBER
 CUBE_OF_NUMBER                                     NUMBER
```

```
-- 3. Create the following Tables maintaining proper Integrity Constraints.
-- Student (s_roll, s_name, s_address, c_id)
-- Course (c_id, c_name, c_fees, c_startdate)

--
 Insert at least 5 records in each table. Keep proper validation so that the value of cou
rse fees (c_fees) lies
-- between 5000-50000 and the c-id starts with the letters 'CR'. [5]
--
 a) Write a PL/SQL code using cursor to increase the course fees of the course 'Python Pr
ogramming' by
--
 10% and other courses by 5%. Ensure that the updation is properly done within the valida
tion limit.
-- b) Write a procedure/function to input the c-
id of a Course and return the Course details. Use proper
-- Exception Handling in case of invalid data input.

/*
Sir I already had Student and Course tables and they are needed to me. So I used
students_sxc_2k21 and courses_sxc_2k21 in place of them.
*/

create table students_sxc_2k21(s_roll number(4) not null, s_name varchar2(15) not null,
    s_address varchar2(25), c_id varchar2(5), primary key(s_roll),
        foreign key(c_id) references courses_sxc_2k21(c_id));

create table courses_sxc_2k21(c_id varchar2(5) not null check(c_id like 'CR%'),
    c_name varchar2(25) not null, c_fees number(5) not null check(c_fees >= 5000 and c_fe
es <= 50000),
        c_startdate date, primary key(c_id));

insert into students_sxc_2k21 values(50,'Manjistha','Kharagpur','CR1');
insert into students_sxc_2k21 values(51,'Chayan','Sodpur','CR1');
insert into students_sxc_2k21 values(52,'Reshav','Sodpur','CR2');
insert into students_sxc_2k21 values(53,'Ravindrababu','Chennai','CR4');
insert into students_sxc_2k21 values(54,'Aswini','Uluberia','CR5');

insert into courses_sxc_2k21 values('CR1', 'Algo Ds', 5500, '02-sep-05');
insert into courses_sxc_2k21 values('CR2', 'Python Programming', 7500, '05-sep-05');
insert into courses_sxc_2k21 values('CR3', 'Soft Eng', 10500, '06-sep-05');
insert into courses_sxc_2k21(c_id, c_name, c_fees) values('CR4', 'Dbms', 9500);
insert into courses_sxc_2k21 values('CR5', 'Java', 15500, '08-sep-05');
```

```
insert into courses_sxc_2k21(c_id, c_name, c_fees) values('CR6', 'Dbms', 49990);

i>

declare
        new_fees courses_sxc_2k21.c_fees%type;
        fees courses_sxc_2k21.c_fees%type;
        name courses_sxc_2k21.c_name%type;
        id courses_sxc_2k21.c_id%type;
        fees_overflow exception;
        start_date courses_sxc_2k21.c_startdate%type;
        cursor cur_courses
        is
        select * from courses_sxc_2k21;
begin
    open cur_courses;
    loop
        fetch cur_courses into id, name, fees, start_date;
        if cur_courses%notfound then
            exit;
        elsif name = 'Python Programming’ then
            new_fees := fees + (10 / 100 * fees);
        else
            new_fees := fees + (5 / 100 * fees);
        end if;
        if new_fees > 50000 then
            raise fees_overflow;
        end if;
        update courses_sxc_2k21
        set c_fees=new_fees
        where c_id = id;
        commit;
    end loop;
    exception
        when fees_overflow then
            dbms_output.put_line('Increment in fees not allowed for id : '|| id || ' coz
it has fees : ' || fees);
    close cur_courses;
end;
/

ii>

set serveroutput on;
declare
    details courses_sxc_2k21%rowtype;
    id courses_sxc_2k21.c_id%type;
    procedure get_details(id in courses_sxc_2k21.c_id%type, details out courses_sxc_2k21%
rowtype)
    is
```

```
    begin
        select * into details from courses_sxc_2k21 where c_id = id;
        exception
            when no_data_found then
                dbms_output.put_line('No such data present for this id value');
    end;
begin
    id := '&course_id';
    get_details(id, details);
    dbms_output.put_line(details.c_id || ' ' || details.c_name || ' ' || details.c_fees |
| ' ' || details.c_startdate);
end;
/
```

```
    8          exception
    9              when no_data_found then
   10                  dbms_output.put_line('No such data present for this id valu
   11      end;
   12  begin
   13      id := '&course_id';
   14      get_details(id, details);
   15      dbms_output.put_line(details.c_id || ' ' || details.c_name || ' ' || de
   16  end;
   17  /
  Enter value for course_id: CR0
  No such data present for this id value

  PL/SQL procedure successfully completed.

  SQL>
```

```
SQL> set serveroutput on;
SQL> declare
  2     details courses_sxc_2k21%rowtype;
  3     id courses_sxc_2k21.c_id%type;
  4     procedure get_details(id in courses_sxc_2k21.c_id%type, details out courses_sxc_2k21%rowtype)
  5     is
  6     begin
  7         select * into details from courses_sxc_2k21 where c_id = id;
  8         exception
  9             when no_data_found then
 10                 dbms_output.put_line('No such data present for this id value');
 11     end;
 12  begin
 13     id := '&course_id';
 14     get_details(id, details);
 15     dbms_output.put_line(details.c_id || ' ' || details.c_name || ' ' || details.c_fees || ' ' || details.c_startdate);
 16  end;
 17  /
Enter value for course_id: CR2
CR2 Python Programming 8250 05-SEP-05
```

```
 29      end loop;
 30       exception
 31           when fees_overflow then
 32               dbms_output.put_line('Increment in fees not allowed fo
 33       close cur_courses;
 34  end;
 35  /

PL/SQL procedure successfully completed.

SQL> select * from courses_sxc_2k21;

C_ID  C_NAME                     C_FEES C_STARTDA
----- ------------------------ ---------- ---------
CR1   Algo Ds                      5775 02-SEP-05
CR2   Python Programming           8250 05-SEP-05
CR3   Soft Eng                    11025 06-SEP-05
CR4   Dbms                         9975
CR5   Java                        16275 08-SEP-05

SQL>
```

```
SQL> select * from courses_sxc_2k21;

C_ID  C_NAME                    C_FEES C_STARTDA
----- ---------------------- ---------- ---------
CR1   Algo Ds                     5500 02-SEP-05
CR2   Python Programming          7500 05-SEP-05
CR3   Soft Eng                   10500 06-SEP-05
CR4   Dbms                        9500
CR5   Java                       15500 08-SEP-05
CR6   Dbms                       49990

6 rows selected.

SQL> select * from students_sxc_2k21;

    S_ROLL S_NAME          S_ADDRESS                 C_ID
---------- --------------- ------------------------- -----
        50 Manjistha       Kharagpur                 CR1
        51 Chayan          Sodpur                    CR1
        52 Reshav          Sodpur                    CR2
        53 Ravindrababu    Chennai                   CR4
        54 Aswini          Uluberia                  CR5

SQL>
```

```
-- 2. Create the following tables with proper integrity constraints:
-- Emp (e_id, e_name, e_sal, d_id)
-- Dept (d_id, d_name, d_location)
--
 Every employee id must begin with ‘EOM’, the salary range of an employee should be betwe
en 15000
--
 and 150000, and the departments are in one of the following locations: Kolkata, Chennai,
 Bangalore,
-- and Gurgaon. Insert at least 5 records in each table. [5]
--
 i) Write a PL/SQL code using Cursor to increase the salary of all the employees of Chenn
ai by 15% and
--
 decrease the salaries of employees residing in Gurgaon by 5%, setting the maximum and mi
nimum
```

```
-- ceiling as given.
--
 ii) Write a procedure/function to input the id (e_id) of an employee and return the corr
esponding
-- employee details. Use proper Exception Handling in case of invalid data input.

create table employees_sxc_2k21(e_id varchar2(10) not null check(e_id like 'EOM%'),
    e_name varchar2(15) not null,
    e_sal number check(e_sal >= 15000 and e_sal <= 150000),
    d_id varchar2(5), primary key(e_id),
        foreign key(d_id) references departs_sxc_2k21(d_id));

create table departs_sxc_2k21(d_id varchar2(5) not null,
    d_name varchar2(15) not null,
    d_location varchar2(15) check(d_location in('Kolkata', 'Chennai', 'Bangalore', 'Gurga
on')),
    primary key(d_id));

insert into Emp values('EOM1','Manjistha',17500,'D1');
insert into Emp values('EOM2','Chayan',18500,'D2');
insert into Emp values('EOM3','Reshav',19500,'D1');
insert into Emp values('EOM6','Karan',19500,'D5');
insert into Emp values('EOM4','Sagnik',57500,'D3');
insert into Emp values('EOM5','Anu',66500,'D4');

insert into Dept values('D1', 'HR', 'Kolkata');
insert into Dept values('D2', 'Specialist', 'Chennai');
insert into Dept values('D3', 'Tech', 'Bangalore');
insert into Dept values('D4', 'Publish', 'Kolkata');
insert into Dept values('D5', 'Supervising', 'Gurgaon');


i>

declare
        new_salary Emp.e_sal%type;
        id Emp.e_id%type;
        name Emp.e_name%type;
        salar Emp.e_sal%type;
        dept Emp.d_id%type;
        cursor cur_employees_1
        is
        select * from Emp where d_id = (select d_id from Dept where d_location = 'Chennai
');

        cursor cur_employees_2
        is
        select * from Emp where d_id = (select d_id from Dept where d_location = 'Gurgaon
');
begin
```

```
    open cur_employees_1;
    loop
        fetch cur_employees_1 into id, name, salar, dept;
        if cur_employees_1%notfound then
            exit;
        else
            new_salary := salar + (15 / 100 * salar);
        end if;
        if new_salary > 150000 then
            new_salary := 150000;
        end if;
        update Emp
        set e_sal = new_salary
        where e_id = id;
        commit;
    end loop;
    open cur_employees_2;
    loop
        fetch cur_employees_2 into id, name, salar, dept;
        if cur_employees_2%notfound then
            exit;
        else
            new_salary := salar - (5 / 100 * salar);
        end if;
        if new_salary < 15000 then
            new_salary := 15000;
        end if;
        update Emp
        set e_sal = new_salary
        where e_id = id;
        commit;
    end loop;
    close cur_employees_1;
    close cur_employees_2;
end;
/

ii>

set serveroutput on
    declare
        details Emp%rowtype;
        id Emp.e_id%type;
    procedure display_details(id in Emp.e_id%type, details out Emp%rowtype)
    is
    begin
        select * into details from Emp where e_id = id;
        exception
            when no_data_found then
                dbms_output.put_line('No such data present for this id value');
```

```
end;
begin
    id := '&emp_id';
    display_details(id, details);
    dbms_output.put_line(details.e_id || ' ' || details.e_name || ' ' || details.e_sa
l || ' ' || details.d_id);
end;
```

```
SQL> select * from Emp;

E_ID       E_NAME               E_SAL D_ID
---------- --------------- ---------- -----
EOM1       Manjistha            17500 D1
EOM2       Chayan               18500 D2
EOM3       Reshav               19500 D1
EOM4       Sagnik               57500 D3
EOM5       Anu                  66500 D4

SQL> select * from Dept;

D_ID  D_NAME          D_LOCATION
----- --------------- ---------------
D1    HR              Kolkata
D2    Specialist      Chennai
D3    Tech            Bangalore
D4    Publish         Kolkata
D5    Supervising     Gurgaon

/ SQL>
```

```
 47      close cur_employees_1;
 48      close cur_employees_2;
 49  end;
 50  /

PL/SQL procedure successfully completed.

SQL> select * from Emp;

E_ID       E_NAME               E_SAL D_ID
---------- --------------- ---------- -----
EOM1       Manjistha            17500 D1
EOM2       Chayan               21275 D2
EOM3       Reshav               19500 D1
EOM4       Sagnik               57500 D3
EOM5       Anu                  66500 D4
EOM6       Karan                18525 D5

6 rows selected.

SQL>
```

```
 10                  dbms_output.put_line('No such
 11      end;
 12     begin
 13          id := '&emp_id';
 14          display_details(id, details);
 15          dbms_output.put_line(details.e_id ||
 16     end;
 17     /
Enter value for emp_id: EOM6
EOM6 Karan 18525 D5

PL/SQL procedure successfully completed.
```

GR – B

5.

a) #include<stdio.h>

```
#include <omp.h>
#include <sys/types.h>

int main(void)
{
    int i, n, m, R = 0;
    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;
}
```

b) Write a C program using Linux System calls to display the contents of a text file, “ sample.txt”.

ANS:

```
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <sys/errno.h>
#include <unistd.h>
int main(int argc, char *argv[])
{

    char *args = {"sample.txt", NULL};
    int fd;
    int status;
    fd = open(args, O_RDONLY); //opening the file sample.txt
                               //if file open is successful, reading the file content and
 displaying it
    if (fd != -1)
    {
        while ((read(fd, &read_byte, sizeof(read_byte))) > 0)
```

```
        {
            write(STDOUT_FILENO, &read_byte, sizeof(read_byte));
        }
        write(STDOUT_FILENO, "File content read successfully.\n", 32);
        close(fd);
    }
    else
    {
        write(STDOUT_FILENO, &errno, sizeof(errno));
        perror("File couldnt be opened.");
        exit(errno);
    }
    exit(1);
}
}
```

7.a) Write a C program using Linux System calls to open a text file, “sample.txt”. The program should
then count the number of vowels present in the file along with its size and display the same.

ANS:

```
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>

int main(int argc, char *argv[])
{
    char buf;
    int size, fd;
    int count = 0;
    fd = open("sample.txt", O_RDONLY); //opening the file in read only mode.
    size = lseek(fd, -1, SEEK_END);    //capturing the size of the file
    printf("The size of the file is:%d", size);

    //reading the file in reverse and checking if there is any vowel or not
       while (size-- >= 0)
    {
        ch = read(fd, &buf, 1);
        if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u' || ch == 'A' ||
 ch == 'E' || ch == 'I' || ch == 'O' || ch == 'U')
            count += 1;          //counting the number of vowels
        lseek(fd, -2, SEEK_CUR); //pointer is moved back by two positions
    }
    printf("The number of vowels present are:%d", count);
    return 0;
}
```

b) Write a C program using OpenMP features to create two parallel threads. One thread should insert an
element into a queue, whereas the other should remove an element from the same queue.
ANS:

```c
#include <stdio.h>
#include <omp.h>
#include <sched.h>
//PROBLEM MAY BE IMPLEMENTED AS PRODUCER CONSUMER PROBLEM WHERE PRODUCES ADDS ELEMENTS IN TO THE QUEUE AND CONSUMER REMOVES
int main()
{
    int Q[50], front = 0, rear = -1, count = 0;
    int id, d, ins = 0;
    omp_set_dynamic(0);
#pragma omp parallel num_threads(2)
    {
        id = omp_get_thread_num();
        if (id == 0) /*Producer*/
            while (1)
            {
#pragma omp critical
                {
                    if (count < 50)
                    {
                        rear = (rear + 1) % 50;
                        ins++;
                        Q[rear] = ins;
                        printf("Inserting element thread %d\n", ins);
                        count++;
                    }
                    else
                    {
                        sched_yield();
                        printf("Queue overflow : max size reached\n");
                    }
                    fgetc(stdin);
                }
            }
        else
        {
            while (1) /*Consumer*/
            {
#pragma omp critical
                {
                    if (count != 0)
                    {
                        d = Q[front];
                        front = (front + 1) % 50;
                        printf("Deleting item thread %d\n", d);
                        count--;
```

```
                    }
                    else
                    {
                        sched_yield();
                        printf("Queue underflow : no item to remove\n");
                    }
                    fgetc(stdin);
                }
            }
        }
    }
}
```

```
  Queue underflow : no item to remove

  Queue underflow : no item to remove

  Queue underflow : no item to remove

  Queue underflow : no item to remove

  Inserting element thread 1

  Inserting element thread 2

  Inserting element thread 3

  Inserting element thread 4

  Inserting element thread 5
Inserting element thread 48

Inserting element thread 49

Inserting element thread 50

Queue overflow : max size reached

Queue overflow : max size reached

Queue overflow : max size reached

Queue overflow : max size reached
```

8.a) Write a C program using Linux System calls to create a child process. The child process should create
a text file, “sample.txt” by accepting input from the user. The parent should display the contents of the
file, “sample.txt” created by the child.

ANS:

```c
//child process to create a text file by accepting characters from the user.
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>
#include <sys/errno.h>
int main(int argc, char *argv[])
{
    int fd;
    char read_byte;
    fd = open("sample.txt", O_WRONLY | O_CREAT | O_EXCL, 0777);
    //if file opened successfully, writing into the file
    if (fd != -1)
    {
        write(STDOUT_FILENO, "Enter the content of the file:\n", 31);
        while ((read(STDIN_FILENO, &read_byte, sizeof(read_byte))) > 0)
        {
            //else writing into the file
            write(fd, &read_byte, sizeof(read_byte));
        }

        write(STDOUT_FILENO, "File creation done!", 20);
        close(fd);
    }
    else
    {
        write(STDOUT_FILENO, &errno, sizeof(errno));
        perror("File open error);
        exit(errno);
    }
    exit(1);
}

//parent process to display the contents of the file

#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <sys/errno.h>
#include <unistd.h>

int main(int argc, char *argv[])
{
    pid_t pid;
```

```
    char *args = {"sample.txt", NULL};
    int fd;
    int status;
    fd = open(args, O_RDONLY); //opening the file sample.txt
    pid = fork();
    if (pid == 0) //child process successfully created
    {
        printf("Entered in child process.Process id:%d,Parent Process ID:%d", getpid(), g
etppid());
        execv("./child1.c", args);
        perror("Failed to replace child process image\n");
        exit(errno);
    }
    else if (pid > 0)
    {
        printf("Returned in parent process.Process id:%d,Child Process:%d", getpid(), pid
);
        //if file open is successful, reading the file content and displaying it
        if (fd != -1)
        {
            while ((read(fd, &read_byte, sizeof(read_byte))) > 0)
            {
                write(STDOUT_FILENO, &read_byte, sizeof(read_byte));
            }
            write(STDOUT_FILENO, "File content read successfully.\n", 32);
            close(fd);
        }
        else
        {
            write(STDOUT_FILENO, &errno, sizeof(errno));
            perror("File couldnt be opened.");
            exit(errno);
        }
        exit(1);
    }
}
```

b)Write a C program using OpenMP features to find the sum of two matrices in linear time.
ANS:

```
#include <stdlib.h>
#include <stdio.h>
#include <omp.h>
void display(int **mat, int order_of_matrix)
{
    int i, j;
    for (i = 0; i < order_of_matrix; i++)
    {
        for (j = 0; j < order_of_matrix; j++)
        {
            printf("%3d ", mat[i][j]);
```

```
        }
        printf("\n");
    }
}
void deallocate_mem(int **mat, int order_of_matrix)
{
    int i;
#pragma omp parallel for shared(mat) private(i)
    for (i = 0; i < order_of_matrix; i++)
    {
        free(mat[i]);
    }
    free(mat);
}
int **sum_of_matrices(int **matrix_1, int **matrix_2, int order_of_matrix)
{
    int **result = NULL;
    int i;
    result = (int **)calloc(order_of_matrix, sizeof(int *));
#pragma omp parallel for shared(result) private(i)
    for (i = 0; i < order_of_matrix; i++)
    {
        result[i] = (int *)calloc(order_of_matrix, sizeof(int));
    }
    for (size_t i = 0; i < order_of_matrix; i++)
    {
#pragma omp parallel for shared(result, matrix_1, matrix_2)
        for (size_t j = 0; j < order_of_matrix; j++)
        {
            result[i][j] = matrix_1[i][j] + matrix_2[i][j];
        }
    }
    return result;
}

int main()
{
    int order_of_matrix_1, order_of_matrix_2;
    int **matrix_1 = NULL;
    int **matrix_2 = NULL;
    int **result = NULL;
    int i, j;
    printf("Enter the order of first matrix : \n");
    scanf("%d", &order_of_matrix_1);
    printf("Enter the order of second matrix : \n");
    scanf("%d", &order_of_matrix_2);
    if (order_of_matrix_1 == order_of_matrix_2)
    {
        omp_set_dynamic(0);
        omp_set_num_threads(order_of_matrix_1);
```

```
        matrix_1 = (int **)calloc(order_of_matrix_1, sizeof(int *));
#pragma omp parallel for shared(matrix_1) private(i)
        for (i = 0; i < order_of_matrix_1; i++)
        {
            matrix_1[i] = (int *)calloc(order_of_matrix_1, sizeof(int));
        }
        for (i = 0; i < order_of_matrix_1; i++)
        {
            for (j = 0; j < order_of_matrix_1; j++)
            {
                printf("matrix[%d][%d] = ", i + 1, j + 1);
                scanf("%d", &matrix_1[i][j]);
            }
        }
        matrix_2 = (int **)calloc(order_of_matrix_2, sizeof(int *));
#pragma omp parallel for shared(matrix_2) private(i)
        for (i = 0; i < order_of_matrix_2; i++)
        {
            matrix_2[i] = (int *)calloc(order_of_matrix_2, sizeof(int));
        }
        for (i = 0; i < order_of_matrix_2; i++)
        {
            for (j = 0; j < order_of_matrix_2; j++)
            {
                printf("matrix[%d][%d] = ", i + 1, j + 1);
                scanf("%d", &matrix_2[i][j]);
            }
        }
        result = sum_of_matrices(matrix_1, matrix_2, order_of_matrix_1);
        printf("\nMatrix 1:- \n\n");
        display(matrix_1, order_of_matrix_1);
        printf("\nMatrix 2:- \n\n");
        display(matrix_2, order_of_matrix_1);
        printf("\nMatrix after summation:- \n\n");
        display(result, order_of_matrix_1);

        deallocate_mem(result, order_of_matrix_1);
        deallocate_mem(matrix_1, order_of_matrix_1);
        deallocate_mem(matrix_2, order_of_matrix_1);
    }
    else
    {
        printf("summation can't be done\n");
    }
    return 0;
}
```

```
matrix[1][3] = 3
matrix[2][1] = 1
matrix[2][2] = 2
matrix[2][3] = 3
matrix[3][1] = 1
matrix[3][2] = 2
matrix[3][3] = 3

Matrix 1:-

  1   2   3
  1   2   3
  1   2   3

Matrix 2:-

  1   2   3
  1   2   3
  1   2   3

Matrix after summation:-

  2   4   6
  2   4   6
  2   4   6
root@kali:~/MscOsLab/msc-os-lab#
```