Practical no: 1

Objective: Introduction to Database Management Systems (DBMS)
Sol: Following is the brief introduction of DBMS with its history and application:-

1. **Overview: DBMS**

A Database Management System (DBMS) is software that enables users to create, manipulate, and manage databases. It provides a systematic way to store, retrieve, and manage data efficiently. DBMS serves as an intermediary between users and the database, ensuring data integrity, security, and consistency.

Key Components of DBMS:

- **Database Engine**: Core service for data storage, processing, and retrieval.
- **Database Schema**: Structure that defines the organization of data.
- **Query Processor**: Interprets and executes database queries.
- **Transaction Management**: Ensures data integrity through concurrent transactions.
- **User Interface**: Tools for users to interact with the database.

2. **SQL (Structured Query Language)**

SQL is the standard language used to communicate with relational databases. It allows users to perform various operations such as:

- **Data Querying**: Retrieving specific data using SELECT statements.
- **Data Manipulation**: Inserting, updating, or deleting records with INSERT, UPDATE, DELETE commands.
- **Data Definition**: Defining database structures with CREATE, ALTER, and DROP commands.

3. **Brief History of DBMS**

The history of DBMS can be traced back to the 1960s when the need for structured data storage arose. Key milestones include:

- **1960s**: Early hierarchical and network databases, such as IBM's Information Management System (IMS).
- **1970**: The introduction of the relational model by Edgar F. Codd, which revolutionized database design and management.
- **1980s**: The emergence of commercial relational database systems like Oracle, IBM DB2, and Microsoft SQL Server.
- **1990s**: The rise of the Internet led to increased demand for database systems, resulting in advancements in scalability and web integration.
- **2000s**: NoSQL databases emerged to handle unstructured data and big data applications.

4. **Applications of DBMS**

DBMS has widespread applications across various industries, including:

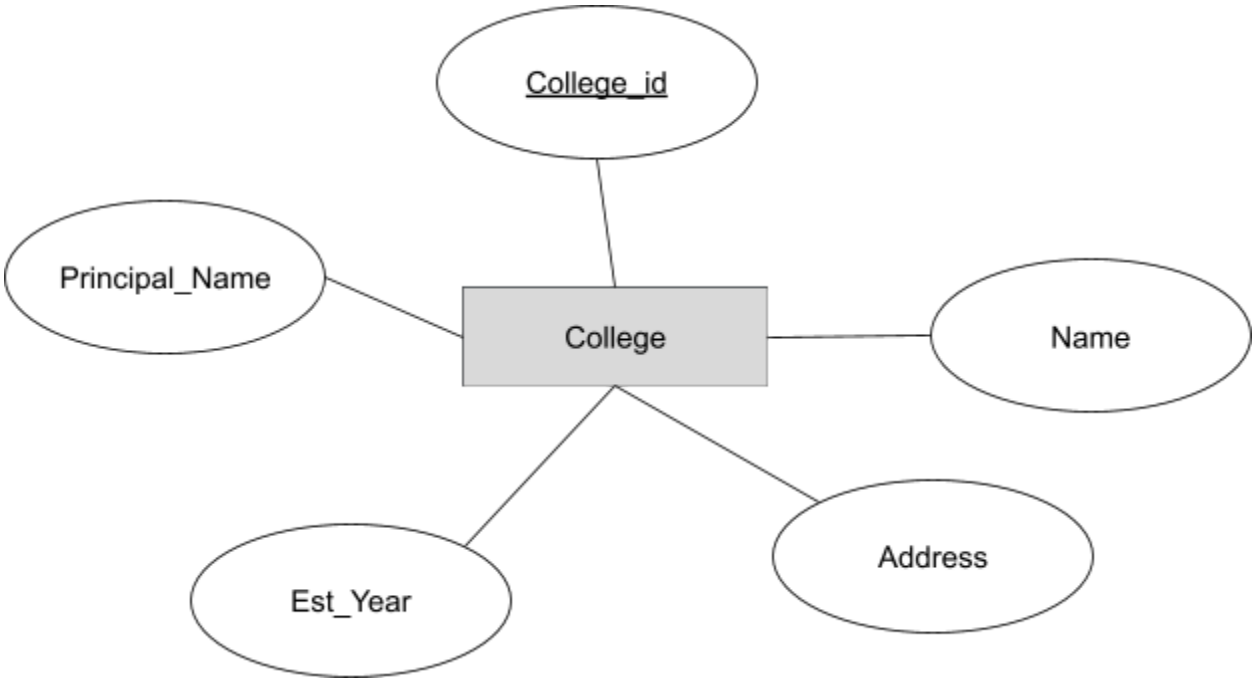
- **Banking**: Managing customer data, transactions, and account details.
- **E-commerce**: Handling product inventories, orders, and customer information.
- **Healthcare**: Storing patient records, treatment histories, and appointment scheduling.
- **Telecommunications**: Managing call records, billing information, and user accounts.
- **Education**: Tracking student records, course registrations, and grades.

Conclusion: Database Management Systems play a critical role in modern data management, enabling organizations to store, retrieve, and manipulate data efficiently. Understanding the fundamental concepts, historical evolution, and diverse applications of DBMS is essential for leveraging its capabilities in various fields.

Practical no: 2

Objective: Draw E-R diagram and convert entities and relationships to relation tables for a given scenario.

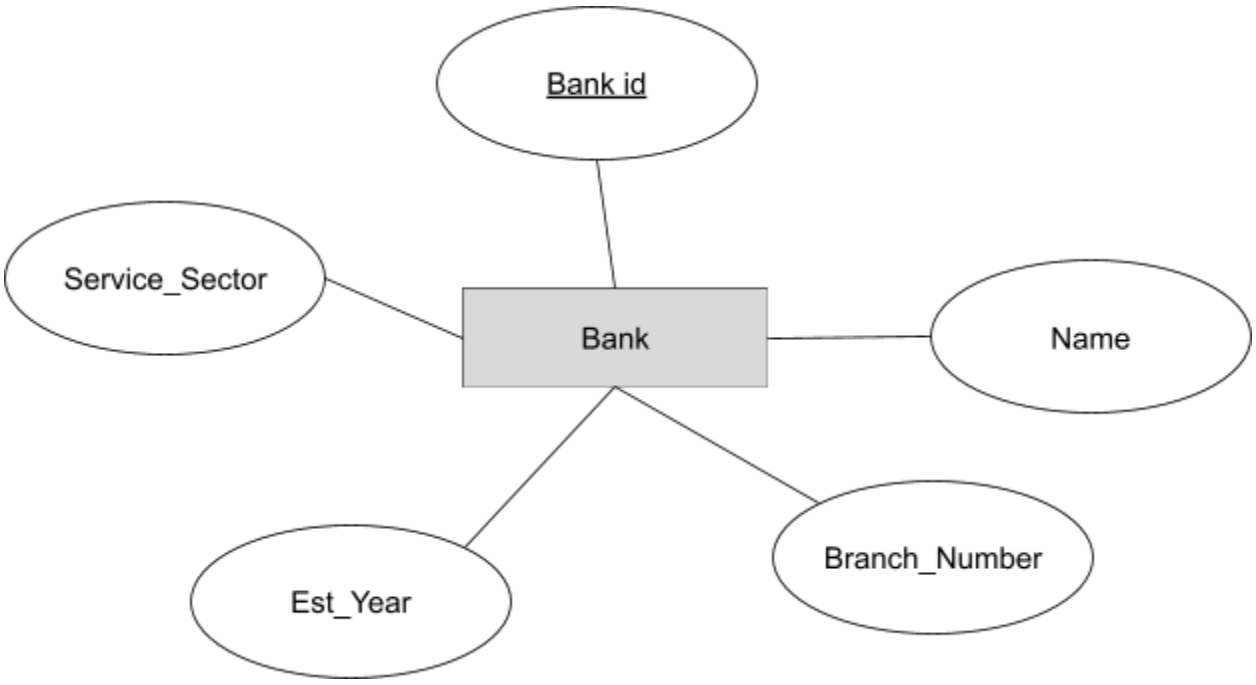
Case I: Following the the ER and relational table for college database:



Following is the Table for same:-

College_id	Name	Est_Year	Principal_Name	Address
1010001	Ambedkar Institute of Technology	1990	Dr. Rajesh Soni	Shakarpur, Delhi
1010002	University School of Automation & Robotics	2000	Dr. Harshdeep Singh	Soorajmal Vihar, Delhi
1010003	NIFT	1996	Dr. Lalita Devi	Delhi Gate, Delhi
1010004	Aloknath Inst. Of technology	2020	Bhanupratap Singh	Faridabad, Haryana

Case II: Following the the ER and relational table for Bank database:



Following is the Table for same:-

Bank_id	Name	Est_Year	Branch_num	Service_Sector
230021	State Bank of India	1990	56	Public
230022	Union Bank of India	2000	23	Public
230024	Indusind Bank	1996	10	Private
230026	Paytm Payments Banks	2020	36	Private

Practical no: 3

Objective:Perform the following:

1. Viewing all databases
2. Creating a Database
3. Viewing all Tables in a Database
4. Creating Tables (With and Without Constraints)
5. Inserting/Updating/Deleting Records in a Table
6. Saving (Commit) and Undoing (rollback)

Sol: Following are the respective queries and tables for the given problem:-

1. View Database: Following is the query:-

```
SHOW DATABASES;
```

```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.00 sec)
```

2. Create Database: Following is the query:-

```
CREATE DATABASE Student;
```

```
mysql> CREATE DATABASE Student; SHOW DATABASES;
Query OK, 1 row affected (0.01 sec)

+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| student            |
| sys                |
+--------------------+
```

3. View tables: Following is the query:-

```
SHOW TABLES;
```

```
mysql> SHOW TABLES;
+------------------+
| Tables_in_student |
+------------------+
| demographic_data  |
| fee_defaulters    |
+------------------+
2 rows in set (0.00 sec)
```

4. Create Tables:

   a. With Constraint: Following is the query:-

```
CREATE TABLE Demographic_Data (
      Roll_No INT PRIMARY KEY,
      Name VARCHAR(20),
      Course VARCHAR(20),
      Semester INT,
      Phone_No INT,
      Address VARCHAR(50)
);
```

```
mysql> CREATE TABLE Demographic_Data (
    -> Roll_No INT PRIMARY KEY,
    -> Name VARCHAR(20),
    -> Course VARCHAR(20),
    -> Semester INT,
    -> Phone_No INT,
    -> Address VARCHAR(50)
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> DESC Demographic_Data
    -> ;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| Roll_No  | int         | NO   | PRI | NULL    |       |
| Name     | varchar(20) | YES  |     | NULL    |       |
| Course   | varchar(20) | YES  |     | NULL    |       |
| Semester | int         | YES  |     | NULL    |       |
| Phone_No | int         | YES  |     | NULL    |       |
| Address  | varchar(50) | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

b. Without Constraint: Following is the query:-

```
CREATE TABLE Fee_Defaulters(
    Roll_No INT,
    Arrears INT
);
```

```
mysql> CREATE TABLE Fee_Defaulters(
    -> Roll_No INT,
    -> Arrears INT
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> DESC Fee_Defaulters;
+---------+------+------+-----+---------+-------+
| Field   | Type | Null | Key | Default | Extra |
+---------+------+------+-----+---------+-------+
| Roll_No | int  | YES  |     | NULL    |       |
| Arrears | int  | YES  |     | NULL    |       |
+---------+------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

5. Insert, Update and Delete: following are the queries:-
   a. Insertion in row:

```
INSERT INTO Demographic_Data( Roll_no, Name, Course,
Semester, Phone_No, Address)
VALUES ( 101, 'Divyanshu Yadav', 'AIDS', 3, 89898989,
'Faridabad'),
(102,'Aditya Kumar', 'AIDS', 3, 78787878, 'Burari'),
(103, 'Abhishek Yadav', 'AIML', 5, 74747474, 'Devali'),
(104, 'Ayush Anjan Dubey','AIML', 5, 89852541, 'Sangam
Vihar');
```

```
mysql> INSERT INTO Demographic_Data( Roll_no, Name, Course, Semester, Phone_No, Address)
    -> VALUES ( 101, 'Divyanshu Yadav', 'AIDS', 3, 89898989, 'Faridabad'),
    -> (102,'Aditya Kumar', 'AIDS', 3, 78787878, 'Burari'),
    -> (103, 'Abhishek Yadav', 'AIML', 5, 74747474, 'Devali'),
    -> (104, 'Ayush Anjan Dubey','AIML', 5, 89852541, 'Sangam Vihar');
Query OK, 4 rows affected (0.01 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> SELECT * FROM Demographic_Data;
+---------+-------------------+--------+----------+----------+--------------+
| Roll_No | Name              | Course | Semester | Phone_No | Address      |
+---------+-------------------+--------+----------+----------+--------------+
|     101 | Divyanshu Yadav   | AIDS   |        3 | 89898989 | Faridabad    |
|     102 | Aditya Kumar      | AIDS   |        3 | 78787878 | Burari       |
|     103 | Abhishek Yadav    | AIML   |        5 | 74747474 | Devali       |
|     104 | Ayush Anjan Dubey | AIML   |        5 | 89852541 | Sangam Vihar |
+---------+-------------------+--------+----------+----------+--------------+
4 rows in set (0.00 sec)
```

b. Updation in row:

```
UPDATE Demographic_Data SET Course='IOT' WHERE Roll_no=101;
```

```
mysql> UPDATE Demographic_Data SET Course='IOT' WHERE Roll_no=101;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM Demographic_Data;
+---------+-------------------+--------+----------+----------+--------------+
| Roll_No | Name              | Course | Semester | Phone_No | Address      |
+---------+-------------------+--------+----------+----------+--------------+
|     101 | Divyanshu Yadav   | IOT    |        3 | 89898989 | Faridabad    |
|     102 | Aditya Kumar      | AIDS   |        3 | 78787878 | Burari       |
|     103 | Abhishek Yadav    | AIML   |        5 | 74747474 | Devali       |
|     104 | Ayush Anjan Dubey | AIML   |        5 | 89852541 | Sangam Vihar |
+---------+-------------------+--------+----------+----------+--------------+
4 rows in set (0.00 sec)
```

c. Deletion in row:

```
DELETE FROM Demographic_Data WHERE Roll_No=101;
```

```
mysql> DELETE FROM Demographic_Data WHERE Roll_No=101;
Query OK, 1 row affected (0.01 sec)

mysql> SELECT * FROM Demographic_Data;
+---------+-------------------+--------+----------+----------+--------------+
| Roll_No | Name              | Course | Semester | Phone_No | Address      |
+---------+-------------------+--------+----------+----------+--------------+
|     102 | Aditya Kumar      | AIDS   |        3 | 78787878 | Burari       |
|     103 | Abhishek Yadav    | AIML   |        5 | 74747474 | Devali       |
|     104 | Ayush Anjan Dubey | AIML   |        5 | 89852541 | Sangam Vihar |
+---------+-------------------+--------+----------+----------+--------------+
3 rows in set (0.00 sec)
```

6. Commit and Rollback: Following are the queries:-

a. Committing the Queries:

```
START TRANSACTION;
INSERT INTO Demographic_Data (Roll_No, Name, Course,
Semester, Phone_No, Address) VALUES(101, 'Divyanshu',
'AIDS', 3, 78587858, 'Faridabad');
COMMIT;
```

```
mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

mysql> INSERT INTO Demographic_Data (Roll_No, Name, Course, Semester, Phone_No, Address)
    -> VALUES(101, 'Divyanshu', 'AIDS', 3, 78587858, 'Faridabad');
Query OK, 1 row affected (0.00 sec)

mysql> COMMIT;
Query OK, 0 rows affected (0.02 sec)

mysql> SELECT * FROM Demographic_Data;
+---------+-------------------+--------+----------+----------+--------------+
| Roll_No | Name              | Course | Semester | Phone_No | Address      |
+---------+-------------------+--------+----------+----------+--------------+
|     101 | Divyanshu         | AIDS   |        3 | 78587858 | Faridabad    |
|     102 | Aditya Kumar      | AIDS   |        3 | 78787878 | Burari       |
|     103 | Abhishek Yadav    | AIML   |        5 | 74747474 | Devali       |
|     104 | Ayush Anjan Dubey | AIML   |        5 | 89852541 | Sangam Vihar |
+---------+-------------------+--------+----------+----------+--------------+
4 rows in set (0.00 sec)
```

b. Rollback Queries:

```
START TRANSACTION;
UPDATE Demographic_Data SET Name='Shammi Yadav',
Course='IOT' WHERE Roll_No=101;
ROLLBACK;
```

```
mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

mysql> UPDATE Demographic_Data SET Name='Shammi Yadav', Course='IOT' WHERE Roll_No=101;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM Demographic_Data;
+---------+-------------------+--------+----------+----------+--------------+
| Roll_No | Name              | Course | Semester | Phone_No | Address      |
+---------+-------------------+--------+----------+----------+--------------+
|     101 | Shammi Yadav      | IOT    |        3 | 78587858 | Faridabad    |
|     102 | Aditya Kumar      | AIDS   |        3 | 78787878 | Burari       |
|     103 | Abhishek Yadav    | AIML   |        5 | 74747474 | Devali       |
|     104 | Ayush Anjan Dubey | AIML   |        5 | 89852541 | Sangam Vihar |
+---------+-------------------+--------+----------+----------+--------------+
4 rows in set (0.00 sec)

mysql> ROLLBACK;
Query OK, 0 rows affected (0.02 sec)

mysql> SELECT * FROM Demographic_Data;
+---------+-------------------+--------+----------+----------+--------------+
| Roll_No | Name              | Course | Semester | Phone_No | Address      |
+---------+-------------------+--------+----------+----------+--------------+
|     101 | Divyanshu         | AIDS   |        3 | 78587858 | Faridabad    |
|     102 | Aditya Kumar      | AIDS   |        3 | 78787878 | Burari       |
|     103 | Abhishek Yadav    | AIML   |        5 | 74747474 | Devali       |
|     104 | Ayush Anjan Dubey | AIML   |        5 | 89852541 | Sangam Vihar |
+---------+-------------------+--------+----------+----------+--------------+
4 rows in set (0.00 sec)
```

Practical no: 4

Objective:Write relational algebra queries for a given set of relations.

Sol: following are the tables and relation operator examples for the given problem:-

Student_id	Name	Phone_no
101	Ramashakar	752XXXX26
102	Hussain	562XXXX85
103	Naman	981XXXX90
104	Imtiyaj Ali	987XXXX12

Table : Student

Student_id	Name	CGPA
102	Hussain	8.5
103	Naman	7.8
104	Imtiyaj Ali	5.5

Table: Graduate_Student

- **Selection (σ)** : Selects the tuple. Following is the example of Selection operator:-

  σ(CGPA > 6)Graduate_student

  This will result in following table(After using projection operator):-

Student_id	Name	CGPA
102	Hussain	8.5
103	Naman	7.8

- **Projection (π)**: Project columns. Following is the example of projection operator:-

    π(Name, Phone_no)Student

    This will give following relations:-

Name	Phone_no
Ramashakar	752XXXX26
Hussain	562XXXX85
Naman	981XXXX90
Imtiyaj Ali	987XXXX12

- **Union (∪)** : Performs union of the record from expression. Following is the example of Union operator:-

    π(Student_id,Phone_no)Student ∪ π(CGPA)Graduate_student

    This will display following output:

Student_id	Phone_no	CGPA
101	752XXXX26	
102	562XXXX85	8.5
103	981XXXX90	7.8
104	987XXXX12	5.5

- **Set Difference (-)** : Display rows of table not in second table. Following is the example:

    π(Name)Student - π(Name)Graduate_student

    This will display:

Name
Ramashakar

- **Set Intersection (∩)** : Perform intersection operation on rows of two table. Following is the example to show implementation:-

    π(Student_id, Name)Student ∩ π(Name)Graduate_student

    This will give following relations:

Student_id	Name
102	Hussain
103	Naman
104	Imtiyaj Ali

- **Rename(ρ)** : Selects the tuple. Following is the example of Selection operator:-

    ρ(Student_id / Roll_number)Graduate_student

    This will result in following table(After using projection operator):-

Student_id	Name	CGPA
102	Hussain	8.5
103	Naman	7.8
104	Imtiyaj Ali	5.5

Practical no: 6

Objective: Write a program in C to perform stack operation using

1. Array
2. Linked List

Program Codes: Following is the code of this problems in C:-

1. Practical6a.c

```
#include <stdio.h>
#define  MAX 5
// function declarations
void push();
void pop();
void peek();
void display();
// global variable declaration
int stack[MAX], tos = -1, choice = 0;//Top of the stack

void push(){
    // when memory is full
    if(tos == MAX-1){
        printf("\nStack Overflow!");
        return;}
    printf("\nEnter Value to push: ");tos++;
    scanf("%d", &stack[tos]);
    printf("Saved successfully.\n");
}

void pop(){
    // when memory is empty
    if(tos==-1){
        printf("\nStack Underflow!");
        return;}
    printf("\nPopped Value: %d\n", stack[tos]);
    tos--;
}
void peek(){
```

```
    // when stack is empty
    if(tos == -1){
        printf("\nStack is empty\n");
        return;}
    printf("\nValue at top: %d\n", stack[tos]);
}

void  display(){
    printf("\n");
    for(int i = tos; i >= 0; i--){
        printf("%4d", stack[i]);}
    printf("\n");
}

int main(){
    while (1){
        printf("\nFollwing stack operations are available:"
                "\n0 → Exit"
                "\n1 → Push data"
                "\n2 → Pop data"
                "\n3 → Peek at top data"
                "\n4 → Display all data"
                "\nEnter your choice: ");
        scanf("%d",&choice);
        switch (choice){
        case 1:
            push();
            break;
        case 2:
            pop();
            break;
        case 3:
            peek();
            break;
        case 4:
            display();
            break;
        case 0:
            return 0;
        default:
            printf("Wrong Choice, retry!\n");
            break;}}}
```

2. Practical6b.c

```
#include <stdio.h>
#include<stdlib.h>
typedef struct node node;
// list declaration
struct node{
    int data;
    struct node* link;
};
// function declaration
void push();
void pop();
void peek();
void display();
// global variable
int choice = 0; node* head= NULL;

void push(){
    node *temp = (node*)malloc(sizeof(node));
    temp→link = NULL;
    printf("\nEnter Data to push:");
    scanf("%d", &temp→data);
    // creation of first node
    if(head == NULL){
        head = temp;
        return;}
    temp→link = head;
    head = temp;
}

void pop(){
    if(head == NULL){
        printf("\nStack Underflow!");
        return;}
    node* temp = head;
    head = head→link;
    printf("\nPopped Data: %d", temp→data);
    free(temp);
}

void peek(){
```

```
    if(head ==NULL){
        printf("\nStack is Empty");
        return;}
    printf("\nValue at top is: %d", head→data);
}

void display(){
    node* temp = head;
    printf("\nStack contains:\n");
    while(temp≠NULL){
        printf("%4d", temp→data);
        temp=temp→link;}
}

int main(){
    while (1){
        printf("\n\nFollwing stack operations are available:"
                "\n0 → Exit"
                "\n1 → Push data"
                "\n2 → Pop data"
                "\n3 → Peek at top data"
                "\n4 → Display all data"
                "\nEnter your choice: ");
        scanf("%d",&choice);
        switch (choice){
        case 1:
            push();
            break;
        case 2:
            pop();
            break;
        case 3:
            peek();
            break;
        case 4:
            display();
            break;
        case 0:
            return 0;
        default:
            printf("Wrong Choice, retry!\n");
            break;}}}
```

Output: Following is the output of the program:-

1. Using Array

```
D:\DV\DSA>gcc Practical6a.c && a.exe

Follwing stack operations are available:
0 -> Exit
1 -> Push data
2 -> Pop data
3 -> Peek at top data
4 -> Display all data
Enter your choice: 1

Enter Value to push: 101
Saved succesfully.

Follwing stack operations are available:
0 -> Exit
1 -> Push data
2 -> Pop data
3 -> Peek at top data
4 -> Display all data
Enter your choice: 1

Enter Value to push: 102
Saved succesfully.

Follwing stack operations are available:
0 -> Exit
1 -> Push data
2 -> Pop data
3 -> Peek at top data
4 -> Display all data
Enter your choice: 1

Enter Value to push: 103
Follwing stack operations are available:
0 -> Exit
1 -> Push data
2 -> Pop data
3 -> Peek at top data
4 -> Display all data
Enter your choice: 1

Enter Value to push: 104
Saved succesfully.

Follwing stack operations are available:
0 -> Exit
1 -> Push data
2 -> Pop data
3 -> Peek at top data
4 -> Display all data
Enter your choice: 1

Enter Value to push: 105
Saved succesfully.

Follwing stack operations are available:
0 -> Exit
1 -> Push data
2 -> Pop data
3 -> Peek at top data
4 -> Display all data
Enter your choice: 1

Stack Overflow!
```

```
Follwing stack operations are available:
0 -> Exit
1 -> Push data
2 -> Pop data
3 -> Peek at top data
4 -> Display all data
Enter your choice: 4

 103 102 101
```

```
Follwing stack operations are available:
0 -> Exit
1 -> Push data
2 -> Pop data
3 -> Peek at top data
4 -> Display all data
Enter your choice: 3

Value at top: 103
```

```
Follwing stack operations are available:
0 -> Exit
1 -> Push data
2 -> Pop data
3 -> Peek at top data
4 -> Display all data
Enter your choice: 2

Popped Value: 103

Follwing stack operations are available:
0 -> Exit
1 -> Push data
2 -> Pop data
3 -> Peek at top data
4 -> Display all data
Enter your choice: 2

Popped Value: 102

Follwing stack operations are available:
0 -> Exit
1 -> Push data
2 -> Pop data
3 -> Peek at top data
4 -> Display all data
Enter your choice: 2

Popped Value: 101

Follwing stack operations are available:
0 -> Exit
1 -> Push data
2 -> Pop data
3 -> Peek at top data
4 -> Display all data
Enter your choice: 2

Stack Underflow!
```

2. Using linked list

```
D:\DV\DSA>gcc Practical6b.c && a.exe


Follwing stack operations are available:
0 -> Exit
1 -> Push data
2 -> Pop data
3 -> Peek at top data
4 -> Display all data
Enter your choice: 1

Enter Data to push:111


Follwing stack operations are available:
0 -> Exit
1 -> Push data
2 -> Pop data
3 -> Peek at top data
4 -> Display all data
Enter your choice: 1

Enter Data to push:112


Follwing stack operations are available:
0 -> Exit
1 -> Push data
2 -> Pop data
3 -> Peek at top data
4 -> Display all data
Enter your choice: 4

Stack contains:
 112 111
```

```
Follwing stack operations are available:
0 -> Exit
1 -> Push data
2 -> Pop data
3 -> Peek at top data
4 -> Display all data
Enter your choice: 3

Value at top is: 112
```

```
0 -> Exit
1 -> Push data
2 -> Pop data
3 -> Peek at top data
4 -> Display all data
Enter your choice: 2

Popped Data: 111

Follwing stack operations are available:
0 -> Exit
1 -> Push data
2 -> Pop data
3 -> Peek at top data
4 -> Display all data
Enter your choice: 2

Stack Underflow!
```