| S.No: 1 | Exp. Name: ***Project Module*** | Date: 2024-06-12 |
|---|---|---|

**Aim**:
Project Module
**Source Code**:

hello.c

```
//write your code here..
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

// Define the structures
typedef struct Book {
    int id;
    char title[100];
    char author[100];
    float price;
    struct Book *next;
} Book;

typedef struct Customer {
    int id;
    char name[100];
    char email[100];
    struct Customer *next;
} Customer;

typedef struct Order {
    int orderId;
    int customerId;
    int bookId;
    struct Order *next;
} Order;

// Head pointers for linked lists
Book *bookHead = NULL;
Customer *customerHead = NULL;
Order *orderHead = NULL;

// Function prototypes
void addBook(int id, char *title, char *author, float price);
void addCustomer(int id, char *name, char *email);
void placeOrder(int orderId, int customerId, int bookId);
void displayBooks();
void displayCustomers();
void displayOrders();

int main() {
    // Sample data
    addBook(1, "The C Programming Language", "Brian W. Kernighan",
30.0);
    addBook(2, "Clean Code", "Robert C. Martin", 25.0);
```

```
    addCustomer(1, "John Doe", "john@example.com");
    addCustomer(2, "Jane Smith", "jane@example.com");

    placeOrder(1, 1, 2);
    placeOrder(2, 2, 1);

    // Display data
    displayBooks();
    displayCustomers();
    displayOrders();

    return 0;
}

// Function definitions
void addBook(int id, char *title, char *author, float price) {
    Book *newBook = (Book *)malloc(sizeof(Book));
    newBook->id = id;
    strcpy(newBook->title, title);
    strcpy(newBook->author, author);
    newBook->price = price;
    newBook->next = bookHead;
    bookHead = newBook;
}

void addCustomer(int id, char *name, char *email) {
    Customer *newCustomer = (Customer *)malloc(sizeof(Customer));
    newCustomer->id = id;
    strcpy(newCustomer->name, name);
    strcpy(newCustomer->email, email);
    newCustomer->next = customerHead;
    customerHead = newCustomer;
}

void placeOrder(int orderId, int customerId, int bookId) {
    Order *newOrder = (Order *)malloc(sizeof(Order));
    newOrder->orderId = orderId;
    newOrder->customerId = customerId;
    newOrder->bookId = bookId;
    newOrder->next = orderHead;
    orderHead = newOrder;
}

void displayBooks() {
    Book *current = bookHead;
```

```
    printf("Books Available:\n");
    while (current != NULL) {
        printf("ID: %d, Title: %s, Author: %s, Price: %.2f\n",
current->id, current->title, current->author, current->price);
        current = current->next;
    }
}

void displayCustomers() {
    Customer *current = customerHead;
    printf("Customers Registered:\n");
    while (current != NULL) {
        printf("ID: %d, Name: %s, Email: %s\n", current->id,
current->name, current->email);
        current = current->next;
    }
}

void displayOrders() {
    Order *current = orderHead;
    printf("Orders Placed:\n");
    while (current != NULL) {
        printf("Order ID: %d, Customer ID: %d, Book ID: %d\n",
current->orderId, current->customerId, current->bookId);
        current = current->next;
    }
}
```

## Execution Results - All test cases have succeeded!

| Test Case - 1 |
| --- |
| **User Output** |
| Hello World |