BANK MANAGEMENT SYSTEM

```
using System;
using System.Collections.Generic;

class Account
{
    public string AccountNumber { get; set; }
    public string AccountHolderName { get; set; }
    public float Balance { get; set; }
    public bool IsActive { get; set; }

    public Account(string accNum = "", string accHolder = "", float bal = 0.0f)
    {
        AccountNumber = accNum;
        AccountHolderName = accHolder;
        Balance = bal;
        IsActive = true; // Initialize isActive to true
    }
}

class Bank
{
    private const int MAX_ACCOUNTS = 100; // Maximum number of accounts
    private List<Account> accounts; // List to store Account objects
    private int count; // Current number of accounts
```

```
public Bank()
{
    accounts = new List<Account>(); // Initialize the list
    count = 0; // Initialize count to 0
}

public void AddAccount(Account account)
{
    if (count < MAX_ACCOUNTS)
    {
        accounts.Add(account); // Add the account to the list
        count++;
        Console.WriteLine("Account added successfully!");
    }
    else
    {
        Console.WriteLine("Bank is full, cannot add more accounts.");
    }
}

public void DisplayAccounts()
{
    if (count == 0)
    {
        Console.WriteLine("No accounts in the bank.");
        return;
    }

    Console.WriteLine("Accounts in the Bank:");
```

```
        foreach (var account in accounts)
        {
            Console.WriteLine($"Account Number: {account.AccountNumber}, Account Holder: 
{account.AccountHolderName}, Balance: ${account.Balance}, Active: {(account.IsActive ? 
"Yes" : "No")}");
        }
    }

    public void SearchAccount(string accountNumber)
    {
        foreach (var account in accounts)
        {
            if (account.AccountNumber == accountNumber)
            {
                Console.WriteLine("Account found:");
                Console.WriteLine($"Account Number: {account.AccountNumber}, Account Holder: 
{account.AccountHolderName}, Balance: ${account.Balance}, Active: {(account.IsActive ? 
"Yes" : "No")}");
                return;
            }
        }
        Console.WriteLine("Account not found.");
    }

    public void ViewTotalBalance(string accountNumber)
    {
        foreach (var account in accounts)
        {
            if (account.AccountNumber == accountNumber)
            {
```

```
            Console.WriteLine($"Total Balance for account {accountNumber} is: 
${account.Balance}");
            return;
          }
        }
        Console.WriteLine("Account not found.");
    }

    public void DepositMoney(string accountNumber, float amount)
    {
        foreach (var account in accounts)
        {
          if (account.AccountNumber == accountNumber)
          {
            if (account.IsActive)
            {
              account.Balance += amount; // Add the deposit amount to balance
              Console.WriteLine($"Deposited ${amount} into account {accountNumber}");
              Console.WriteLine($"New balance: ${account.Balance}");
            }
            else
            {
              Console.WriteLine($"Account {accountNumber} is inactive.");
            }
            return;
          }
        }
        Console.WriteLine("Account not found.");
    }
```

```
public void WithdrawMoney(string accountNumber, float amount)
{
    foreach (var account in accounts)
    {
        if (account.AccountNumber == accountNumber)
        {
            if (account.IsActive)
            {
                if (account.Balance >= amount)
                {
                    account.Balance -= amount; // Subtract the withdrawal amount from balance
                    Console.WriteLine($"Withdrew ${amount} from account {accountNumber}");
                    Console.WriteLine($"New balance: ${account.Balance}");
                }
                else
                {
                    Console.WriteLine($"Insufficient balance in account {accountNumber}");
                }
            }
            else
            {
                Console.WriteLine($"Account {accountNumber} is inactive.");
            }
            return;
        }
    }
    Console.WriteLine("Account not found.");
}
```

```
public void DeactivateAccount(string accountNumber)
{
    foreach (var account in accounts)
    {
        if (account.AccountNumber == accountNumber)
        {
            if (account.IsActive)
            {
                account.IsActive = false; // Mark account as inactive
                Console.WriteLine($"Account {accountNumber} has been deactivated.");
            }
            else
            {
                Console.WriteLine($"Account {accountNumber} is already inactive.");
            }
            return;
        }
    }
    Console.WriteLine("Account not found.");
}

public void ActivateAccount(string accountNumber)
{
    foreach (var account in accounts)
    {
        if (account.AccountNumber == accountNumber)
        {
            if (!account.IsActive)
            {
                account.IsActive = true; // Mark account as active
                Console.WriteLine($"Account {accountNumber} has been activated.");
            }
            else
            {
                Console.WriteLine($"Account {accountNumber} is already active.");
            }
            return;
        }
    }
    Console.WriteLine("Account not found.");
  }

  public int GetAccountCount()
  {
    return count;
  }
}

class Program
{
  static void AdminMenu(Bank bank)
  {
    int choice;
    do
    {
      Console.WriteLine("\nAdmin Menu");
      Console.WriteLine("1. Add Account");
      Console.WriteLine("2. Display Accounts");
Console.WriteLine("3. Deactivate Account");
Console.WriteLine("4. Activate Account");
Console.WriteLine("5. Count Accounts in Bank");
Console.WriteLine("6. Exit");
Console.Write("Enter your choice: ");
choice = int.Parse(Console.ReadLine());

switch (choice)
{
    case 1:
        Console.Write("Enter account number: ");
        string accountNumber = Console.ReadLine();
        Console.Write("Enter account holder name: ");
        string accountHolder = Console.ReadLine();
        Console.Write("Enter initial balance: ");
        float balance = float.Parse(Console.ReadLine());
        bank.AddAccount(new Account(accountNumber, accountHolder, balance));
        break;

    case 2:
        bank.DisplayAccounts();
        break;

    case 3:
        Console.Write("Enter account number to deactivate: ");
        accountNumber = Console.ReadLine();
        bank.DeactivateAccount(accountNumber);
        break;
```

```
            case 4:
                Console.Write("Enter account number to activate: ");
                accountNumber = Console.ReadLine();
                bank.ActivateAccount(accountNumber);
                break;

            case 5:
                Console.WriteLine($"Total accounts in bank: {bank.GetAccountCount()}");
                break;

            case 6:
                Console.WriteLine("Exiting admin menu...");
                break;

            default:
                Console.WriteLine("Invalid choice! Please try again.");
                break;
        }
    } while (choice != 6);
}

static void CustomerMenu(Bank bank)
{
    int choice;
    do
    {
        Console.WriteLine("\nCustomer Menu");
        Console.WriteLine("1. Search Account");
        Console.WriteLine("2. Deposit Money");
Console.WriteLine("3. Withdraw Money");
Console.WriteLine("4. View Total Balance");
Console.WriteLine("5. Exit");
Console.Write("Enter your choice: ");
choice = int.Parse(Console.ReadLine());

switch (choice)
{
  case 1:
    Console.Write("Enter account number to search: ");
    string accountNumber = Console.ReadLine();
    bank.SearchAccount(accountNumber);
    break;

  case 2:
    Console.Write("Enter account number to deposit: ");
    accountNumber = Console.ReadLine();
    Console.Write("Enter amount to deposit: ");
    float amount = float.Parse(Console.ReadLine());
    bank.DepositMoney(accountNumber, amount);
    break;

  case 3:
    Console.Write("Enter account number to withdraw from: ");
    accountNumber = Console.ReadLine();
    Console.Write("Enter amount to withdraw: ");
    amount = float.Parse(Console.ReadLine());
    bank.WithdrawMoney(accountNumber, amount);
    break;
```

```
            case 4:
                Console.Write("Enter account number to view balance: ");
                accountNumber = Console.ReadLine();
                bank.ViewTotalBalance(accountNumber);
                break;

            case 5:
                Console.WriteLine("Exiting customer menu...");
                break;

            default:
                Console.WriteLine("Invalid choice! Please try again.");
                break;
        }
    } while (choice != 5);
}

static void Main(string[] args)
{
    Bank bank = new Bank();
    int userType;

    do
    {
        Console.WriteLine("Welcome to the Bank Management System");
        Console.WriteLine("Select User Type: ");
        Console.WriteLine("1. Admin");
        Console.WriteLine("2. Customer");
```

```
            Console.WriteLine("3. Exit");
            Console.Write("Enter your choice: ");
            userType = int.Parse(Console.ReadLine());

            switch (userType)
            {
                case 1:
                    AdminMenu(bank);
                    break;

                case 2:
                    CustomerMenu(bank);
                    break;

                case 3:
                    Console.WriteLine("Exiting the system...");
                    break;

                default:
                    Console.WriteLine("Invalid user type selected. Please try again.");
                    break;
            }
        } while (userType != 3);
    }
}
```