

Riddhi Jain Task-8

1) Login/Signup System using C++

A-

Login system.cpp

```
1  #include <iostream>
2  #include <fstream>
3  #include <string>
4  using namespace std;
5  bool userExists(const string& username) {
6      ifstream file("users.txt");
7      string line;
8      while (getline(file, line)) {
9          size_t delimiter_pos = line.find(':');
10         if (delimiter_pos != string::npos) {
11             string stored_username = line.substr(0, delimiter_pos);
12             if (stored_username == username) {
13                 return true;
14             }
15         }
16     }
17     return false;
18 }
19 bool authenticateUser(const string& username, const string& password) {
20     ifstream file("users.txt");
21     string line;
22     while (getline(file, line)) {
23         size_t delimiter_pos = line.find(':');
24         if (delimiter_pos != string::npos) {
25             string stored_username = line.substr(0, delimiter_pos);
26             string stored_password = line.substr(delimiter_pos + 1);
27             if (stored_username == username && stored_password == password) {
28                 return true;
29             }
30         }
31     }
32     return false;
33 }
34 void signup() {
35     string username, password;
36     cout << "Enter a username: ";
37     cin >> username;
38     if (userExists(username)) {
39         cout << "Username already exists. Please try again.\n";
40         return;
41     }
42     cout << "Enter a password: ";
43     cin >> password;
44
45     ofstream file("users.txt", ios::app);
46     file << username << ":" << password << "\n";
47     file.close();
48     cout << "Signup successful.\n";
49 }
50 void login() {
51     string username, password;
52     cout << "Enter your username: ";
53     cin >> username;
54     cout << "Enter your password: ";
55     cin >> password;
56
57     if (authenticateUser(username, password)) {
58         cout << "Login successful. Welcome, " << username << "!\n";
59     } else {
60         cout << "Invalid username or password. Please try again.\n";
61     }
62 }
```

```

63 int main() {
64     int choice;
65     while (true) {
66         cout << "1. Signup\n2. Login\n3. Exit\nChoose an option: ";
67         cin >> choice;
68
69         switch (choice) {
70             case 1:
71                 signup();
72                 break;
73             case 2:
74                 login();
75                 break;
76             case 3:
77                 cout << "Exiting...\n";
78                 return 0;
79             default:
80                 cout << "Invalid choice. Please try again.\n";
81         }
82     }
83 }
84

```

Output:-

E:\C++\Login system.exe

```

1. Signup
2. Login
3. Exit
Choose an option: 1
Enter a username: Riddhi146
Enter a password: Riddhi@146
Signup successful.
1. Signup
2. Login
3. Exit
Choose an option: 2
Enter your username: Riddhi146
Enter your password: Riddhi@146
Login successful. Welcome, Riddhi146!
1. Signup
2. Login
3. Exit
Choose an option: 3
Exiting...

-----
Process exited after 916.8 seconds with return value 0
Press any key to continue . . .

```

2) Bookshop Inventory System using C++

A-

[*] Bookshop_inventory.cpp

```
1  #include <iostream>
2  #include <fstream>
3  #include <vector>
4  #include <sstream>
5  using namespace std;
6  struct Book {
7      string title;
8      string author;
9      string isbn;
10     double price;
11     int quantity;
12 };
13 vector<Book> loadInventory(const string& filename) {
14     vector<Book> inventory;
15     ifstream file(filename);
16     string line;
17     while (getline(file, line)) {
18         istringstream iss(line);
19         Book book;
20         getline(iss, book.title, ',');
21         getline(iss, book.author, ',');
22         getline(iss, book.isbn, ',');
23         iss >> book.price;
24         iss.ignore(1, ',');
25         iss >> book.quantity;
26         inventory.push_back(book);
27     }
28     return inventory;
29 }
30 void saveInventory(const vector<Book>& inventory, const string& filename) {
31     ofstream file(filename);
32     for (const auto& book : inventory) {
33         file << book.title << "," << book.author << "," << book.isbn << ","
34             << book.price << "," << book.quantity << "\n";
35     }
36 }
37 void addBook(vector<Book>& inventory) {
38     Book book;
39     cout << "Enter title: ";
40     cin.ignore();
41     getline(cin, book.title);
42     cout << "Enter author: ";
43     getline(cin, book.author);
44     cout << "Enter ISBN: ";
45     cin >> book.isbn;
46     cout << "Enter price: ";
47     cin >> book.price;
48     cout << "Enter quantity: ";
49     cin >> book.quantity;
50     inventory.push_back(book);
51     cout << "Book added successfully.\n";
52 }
53 void removeBook(vector<Book>& inventory) {
54     string isbn;
55     cout << "Enter ISBN of the book to remove: ";
56     cin >> isbn;
57     auto it = remove_if(inventory.begin(), inventory.end(), [&](const Book& book) {
58         return book.isbn == isbn;
59     });
60     if (it != inventory.end()) {
61         inventory.erase(it, inventory.end());
62         cout << "Book removed successfully.\n";
63     } else {
64         cout << "Book not found.\n";
65     }
66 }
```

```

67 void updateBook(vector<Book>& inventory) {
68     string isbn;
69     cout << "Enter ISBN of the book to update: ";
70     cin >> isbn;
71     for (auto& book : inventory) {
72         if (book.isbn == isbn) {
73             cout << "Enter new title (or press enter to keep unchanged): ";
74             cin.ignore();
75             string newTitle;
76             getline(cin, newTitle);
77             if (!newTitle.empty()) book.title = newTitle;
78             cout << "Enter new author (or press enter to keep unchanged): ";
79             string newAuthor;
80             getline(cin, newAuthor);
81             if (!newAuthor.empty()) book.author = newAuthor;
82             cout << "Enter new price (or press enter to keep unchanged): ";
83             string newPrice;
84             getline(cin, newPrice);
85             if (!newPrice.empty()) book.price = stod(newPrice);
86             cout << "Enter new quantity (or press enter to keep unchanged): ";
87             string newQuantity;
88             getline(cin, newQuantity);
89             if (!newQuantity.empty()) book.quantity = stoi(newQuantity);
90             cout << "Book updated successfully.\n";
91             return;
92         }
93     }
94     cout << "Book not found.\n";
95 }

96 void displayInventory(const vector<Book>& inventory) {
97     if (inventory.empty()) {
98         cout << "No books in inventory.\n";
99         return;
100     }
101     for (const auto& book : inventory) {
102         cout << "Title: " << book.title << "\n"
103             << "Author: " << book.author << "\n"
104             << "ISBN: " << book.isbn << "\n"
105             << "Price: $" << book.price << "\n"
106             << "Quantity: " << book.quantity << "\n"
107             << "-----\n";
108     }
109 }

110 int main() {
111     vector<Book> inventory = loadInventory("inventory.txt");
112     int choice;
113     while (true) {
114         cout << "1. Add Book\n2. Remove Book\n3. Update Book\n4. Display Inventory\n5. Exit\nChoose an option: ";
115         cin >> choice;
116
117         switch (choice) {
118             case 1:
119                 addBook(inventory);
120                 break;
121             case 2:
122                 removeBook(inventory);
123                 break;
124             case 3:
125                 updateBook(inventory);
126                 break;
127             case 4:
128                 displayInventory(inventory);
129                 break;
130             case 5:
131                 saveInventory(inventory, "inventory.txt");
132                 cout << "Exiting...\n";
133                 return 0;
134             default:
135                 cout << "Invalid choice. Please try again.\n";
136         }
137     }
138 }
139

```

3) Sudoku Game using C++

A-

```
Sudoku.cpp
1  #include <iostream>
2  #include <vector>
3  #define N 9
4  class Sudoku {
5  private:
6      int board[N][N];
7      bool isValid(int row, int col, int num) {
8          for (int x = 0; x < N; x++) {
9              if (board[row][x] == num || board[x][col] == num || board[row - row % 3 + x / 3][col - col % 3 + x % 3] == num) {
10                 return false;
11             }
12         }
13         return true;
14     }
15     bool isSolved() {
16         for (int i = 0; i < N; i++) {
17             for (int j = 0; j < N; j++) {
18                 if (board[i][j] == 0) {
19                     return false;
20                 }
21             }
22         }
23         return true;
24     }
25 public:
26     Sudoku() {
27         int initialBoard[N][N] = {
28             {5, 3, 0, 0, 7, 0, 0, 0, 0},
29             {6, 0, 0, 1, 9, 5, 0, 0, 0},
30             {0, 9, 8, 0, 0, 0, 0, 6, 0},
31             {8, 0, 0, 0, 6, 0, 0, 0, 3},
32             {4, 0, 0, 8, 0, 3, 0, 0, 1},
33             {7, 0, 0, 0, 2, 0, 0, 0, 6},
34             {0, 6, 0, 0, 0, 0, 2, 8, 0},
35             {0, 0, 0, 4, 1, 9, 0, 0, 5},
36             {0, 0, 0, 0, 8, 0, 0, 7, 9}
37         };
38         for (int i = 0; i < N; i++) {
39             for (int j = 0; j < N; j++) {
40                 board[i][j] = initialBoard[i][j];
41             }
42         }
43     }
44     void displayBoard() {
45         for (int i = 0; i < N; i++) {
46             if (i % 3 == 0 && i != 0) {
47                 std::cout << "-----\n";
48             }
49             for (int j = 0; j < N; j++) {
50                 if (j % 3 == 0 && j != 0) {
51                     std::cout << " | ";
52                 }
53                 std::cout << board[i][j] << " ";
54             }
55             std::cout << "\n";
56         }
57     }
```

```

58     bool makeMove(int row, int col, int num) {
59         if (board[row][col] == 0 && isValid(row, col, num)) {
60             board[row][col] = num;
61             return true;
62         }
63         return false;
64     }
65
66     bool gameWon() {
67         return isSolved();
68     }
69 };
70 int main() {
71     Sudoku game;
72     int row, col, num;
73     while (!game.gameWon()) {
74         game.displayBoard();
75         std::cout << "Enter your move (row col num): ";
76         std::cin >> row >> col >> num;
77
78         if (row < 0 || col < 0 || num < 0) {
79             std::cout << "Exiting game...\n";
80             break;
81         }
82
83         if (game.makeMove(row, col, num)) {
84             std::cout << "Move accepted.\n";
85         } else {
86             std::cout << "Invalid move. Try again.\n";
87         }
88     }
89     if (game.gameWon()) {
90         std::cout << "Congratulations! You've solved the Sudoku puzzle.\n";
91     }
92     return 0;
93 }

```

Output:-

```

E:\C++\Sudoku.exe
5 3 0 | 0 7 0 | 0 0 0
6 0 0 | 1 9 5 | 0 0 0
0 9 8 | 0 0 0 | 0 6 0
-----
8 0 0 | 0 6 0 | 0 0 3
4 0 0 | 8 0 3 | 0 0 1
7 0 0 | 0 2 0 | 0 0 6
-----
0 6 0 | 0 0 0 | 2 8 0
0 0 0 | 4 1 9 | 0 0 5
0 0 0 | 0 8 0 | 0 7 9
Enter your move (row col num): 3
5
8
Invalid move. Try again.
5 3 0 | 0 7 0 | 0 0 0
6 0 0 | 1 9 5 | 0 0 0
0 9 8 | 0 0 0 | 0 6 0
-----
8 0 0 | 0 6 0 | 0 0 3
4 0 0 | 8 0 3 | 0 0 1
7 0 0 | 0 2 0 | 0 0 6
-----
0 6 0 | 0 0 0 | 2 8 0
0 0 0 | 4 1 9 | 0 0 5
0 0 0 | 0 8 0 | 0 7 9
Enter your move (row col num): _

```