# Programming Fundamental

Programming Day - Week 10

## Introduction

Welcome to your favorite day of the week which is programming day🎉. This week, we shall work together to learn and implement new programming concepts.

**Let's do some coding.**

## University Admission Management System

Students, Let's create the first version of the UAMS that has two roles

- User
- Admin

Notice the following .gif for better understanding


Lets write down the functions are arrays that will be required for developing this program.

- We need three global arrays for storing name, password, and roles of the users.
- In addition we need a global variable to count the number of students added in these arrays.
- Lastly, we need different single responsbility functions for implementing the logic of UAMS.

```
const int userArrSize = 10;
string users[userArrSize];
string passwords[userArrSize];
string roles[userArrSize];

int usersCount = 0;

  string signIn(string name, string password);
  bool signUp(string name, string password, string role);
  void topHeader();
  void subMenuBeforeMainMenu(string submenu);
  void subMenu(string submenu);
  int loginMenu();
  int adminMenu();
  void adminInterface();
  int userMenu();
  void userInterface();
  void clearScreen();
```

Lets define these functions one by one according to their functionality.

- The first menu that will return the choice of the user as an interger number.

```
int loginMenu()
{
    int option;
    cout << "1. SignIn with your Credentials" << endl;
    cout << "2. SignUp to get your Credentials" << endl;
    cout << "3. Exit the Application" << endl;
    cout << endl;
    cout << "Enter the Option Number > ";
    cin >> option;
    return option;
}
```

- The signIn function that will return the role of the user and take two input parameters as name and password.

```
string signIn(string name, string password)
{
    for (int index = 0; index < usersCount; index++)
    {
        if (users[index] == name && passwords[index] == password)
        {
            return roles[index];
        }
    }
    return "undefined";
}
```

- The signUp function that will return a boolean value after taking in input parameters name, password and role of the user.

```
bool signUp(string name, string password, string role)
{
    bool isPresent = false;

    for (int index = 0; index < usersCount; index++)
    {
        if (users[index] == name && passwords[index] == password)
        {
            isPresent = true;
            break;
        }
    }
    if (isPresent == true)
    {
        return 0;
        cout << "User Already Present";
    }
```

```
    else if (usersCount < userArrSize)
    {
        users[usersCount] = name;
        passwords[usersCount] = password;
        roles[usersCount] = role;
        usersCount++;
        return true;
    }
    else
    {
        return false;
    }
}
```

- topHeader function will be used to print the main header of the UAMS.

```
void topHeader()
{
    cout << "*************************************************************" << endl;
    cout << "*        University Admission Management System             *" << endl;
    cout << "*************************************************************" << endl;
    cout << endl;
}
```

- The subMenuBeforeMainMenu will print the submenu

```
void subMenuBeforeMainMenu(string submenu)
{
    string message = submenu + " Menu";
    cout << message << endl;
    cout << "---------------------" << endl;
}
```

- The subMenu will print the submenu

```
void subMenu(string submenu)
{
    string message = "Main Menu > " + submenu;
    cout << message << endl;
    cout << "---------------------" << endl;
}
```

- clearScreen function will clear the screen.

```
void clearScreen()
{
    cout << "Press Any Key to Continue.." << endl;
    getch();
    system("cls");
}
```

- adminMenu will print the admin panel on the console.

```
int adminMenu()
{
    int option;
    cout << "Select one of the following options number..." << endl;
    cout << "1. Add New Student Record" << endl;
    cout << "2. View all Records" << endl;
    cout << "3. Print the Record in Merit Order" << endl;
    cout << "4. Generate Merit List" << endl;
    cout << "5. Seat Management" << endl;
    cout << "6. Exit" << endl;
    cout << "Your Option..";
    cin >> option;
    return option;
}
```

- adminInterface will be the controller function for admin panel

```
void adminInterface()
{
    int adminOption = 0;
    while (adminOption != 6)
    {
        topHeader();
        subMenu("");
        adminOption = adminMenu();
        if (adminOption == 1)
        {
            system("cls");
            topHeader();
            subMenu("Add Student");
            // Implement the Add Student Functionality
        }
        // Implement the rest of the Admin Options
        clearScreen();
    }
}
```

- userMenu will print the user menu on the console screen.

```
int userMenu()
{
    int option;
    cout << "Select one of the following options number..." << endl;
    cout << "1. View your Details" << endl;
    cout << "2. Update password" << endl;
    cout << "3. Apply in a Degree Program" << endl;
    cout << "4. Exit" << endl;
    cout << "Your Option..";
    cin >> option;
    return option;
}
```

- userInterface is the driver function for the user panel

```
void userInterface()
{
    int userOption = 0;
    while (userOption != 4)
    {
        topHeader();
        subMenu("");
        userOption = userMenu();
        if (userOption == 1)
        {
            system("cls");
            topHeader();
            subMenu("View Details");
            // Implement the View Details Functionality
        }
        // Implement the rest of the User Options
        clearScreen();
    }
}
```

Let's now implement the Main() driver function.

```
main()
{
    int loginOption = 0;
    while (loginOption != 3)
    {
        topHeader();
        subMenuBeforeMainMenu("Login");
        loginOption = loginMenu();
        if (loginOption == 1)
        {
            system("cls");

            string name;
            string password;
            string role;
            topHeader();
            subMenuBeforeMainMenu("SignIn");
            cout << "Enter your Name: " << endl;
            cin >> name;
            cout << "Enter your Password: " << endl;
            cin >> password;
            role = signIn(name, password);
```

```
            if (role == "Admin")
            {
                clearScreen();
                adminInterface();
            }
            else if (role == "User")
            {
                clearScreen();
                userInterface();
            }
            else if (role == "undefined")
            {
                cout << "You Entered wrong Credentials" << endl;
            }
        }
        else if (loginOption == 2)
        {
            system("cls");
            string name;
            string password;
            string role;
            topHeader();
            subMenuBeforeMainMenu("SignUp");
            cout << "Enter your Name: " << endl;
            cin >> name;
            cout << "Enter your Password: " << endl;
            cin >> password;
            cout << "Enter your Role (Admin or User): " << endl;
            cin >> role;
            bool isValid = signUp(name, password, role);
            if (isValid)
            {
                cout << "SignedUp Successfully" << endl;
            }
             if (!isValid)
             {
                 cout << "Users in the System have exceeded the Capacity" << endl;
             }
         }
         clearScreen();
    }
}
```

**Congratulations !! You have implemented your first version of the UAMS.**

**Project Version 01:** Complete your business application with the help from Teaching Assitant assigned to you.

# Programming Fundamental

Programming Day - Week 10

## Bonus Tasks (optional)

**Task 01(CP):**

Create a function that determines whether elements in an array can be re-arranged to form a consecutive list of numbers where each number appears exactly once.

**Test Cases:**

| | | |
|---|---|---|
| Array Length: 5<br>[5, 1, 4, 3, 2] | **True** | // Can be re-arranged to form [1, 2, 3, 4, 5] |
| Array Length: 6<br>[5, 1, 4, 3, 2, 8] | **False** | |
| Array Length: 6<br>[5, 6, 7, 8, 9, 9] | **False** | // 9 appears twice |

**Task 02(CP):**

Given a list of directions to spin, "left" or "right", return an integer of how many full 360° rotations were made. Note that each word in the array counts as a 90° rotation in that direction.

| | | |
|---|---|---|
| ["right", "right", "right", "right", "left", "right"] | 1 | // You spun right 4 times (90 * 4 = 360)<br>// You spun left once (360 - 90 = 270)<br>// But you spun right once more to make a full rotation (270 + 90 = 360) |
| ["left", "right", "left", "right"] | 0 | |
| ["right", "right", "right", "right", "right", "right", "right", "right"] | 2 | |
| ["left", "left", "left", "left"] | 1 | |

**Good Luck and Best Wishes !!**

**Happy Coding ahead :)**

Muhammad Irzam & Maida Shahid, Department of Computer Science, UET Lahore