# C++ Assignment: Movie Ticket Booking System

## Objective:

The goal of this assignment is to help you understand and apply enumeration types, `typedef` statements, namespaces, string operations, and input/output operations in C++. You will create a program that models a simple ticket booking system for a theater.

## Instructions:

1. Define Enumeration Types

- Define an enumeration type `MovieGenre` with the following values: `ACTION`, `COMEDY`, `DRAMA`, `SCI_FI`, and `HORROR`.

- Define another enumeration type `TicketType` with values: `REGULAR`, `VIP`, and `STUDENT`.

- Declare variables of these types within your main function.

2. Using typedef and Anonymous Data Types

- Use `typedef` to create an alias called `String` for the `std::string` type.

- Define an anonymous enum for `SeatRow` with values representing row labels `A`, `B`, `C`, `D`, and `E`. Declare a variable of this type inside your main function.

3. Namespaces

- Place all `std` functions (e.g., `cout`, `cin`, etc.) within a `namespace theater_booking` and reference this namespace in your main function to avoid repeatedly using the `std::` prefix.

4. Functions and Enumeration Types

- Write a function `getGenreName` that takes a `MovieGenre` enumeration value and returns the genre as a string. For example, if the genre is `ACTION`, return `"Action"`.

- Write another function `calculateTicketPrice` that takes a `TicketType` and returns the price as a `double`. Set the following prices:

  - `REGULAR`: $10.00

  - `VIP`: $20.00

  - `STUDENT`: $8.00

5. User Input/Output

- Prompt the user to select a `MovieGenre` (by displaying a list with corresponding numbers, e.g., 1 for `ACTION`, 2 for `COMEDY`, etc.).

- Prompt the user to select a `TicketType` in the same way.

- Output the selected `MovieGenre`, `TicketType`, and calculated ticket price.

6. Relational Operators and Additional String Operations

- Define two string variables using your `String` alias: `userName` and `password`.

- Prompt the user to enter a username and password. Store these in the respective variables.

- Use relational operators to check if the entered username matches `"Your first name "` and if the password matches `"Your Lastname"`. Print a success message if both match; otherwise, print an error.

For example: My first name is Sumanth and last name is Burugula so they should be matched. In your program use your first name and last name.

7. Quick Review - Additional Features

- Extend the program to ask the user if they want to book another ticket. If yes, allow the process to repeat; otherwise, terminate the program.

## Grading Criteria:

- Correctness: The program runs and performs as specified.

- Code Structure: Code is well-organized, with comments where necessary.

- Usage of Enumerations: Proper declaration, assignment, and operations on enumeration types.

- String Operations: Proper use of `String` type, input/output, and relational comparisons.

- Namespaces and typedefs: Correct usage of namespaces and `typedef`.

Submission:

Submit the source code file to git and blackboard along with the screenshot of the output with appropriate comments explaining each part of the code.

Output:

```
Enter username: Sumanth
Enter password: Burugula
Login successful!

Select Movie Genre:
1. Action
2. Comedy
3. Drama
4. Sci-Fi
5. Horror
Enter choice (1-5): 3

Select Ticket Type:
1. Regular
2. VIP
3. Student
Enter choice (1-3): 1

You selected: Drama
Ticket Type: Regular
Ticket Price: $10

Would you like to book another ticket? (y/n): n
Thank you for using the theater booking system!
```

```
Enter username: Sumanth
Enter password: Burugula
Login successful!

Select Movie Genre:
1. Action
2. Comedy
3. Drama
4. Sci-Fi
5. Horror
Enter choice (1-5): 1

Select Ticket Type:
1. Regular
2. VIP
3. Student
Enter choice (1-3): 2

You selected: Action
Ticket Type: VIP
Ticket Price: $20

Would you like to book another ticket? (y/n): y
Enter username: John
Enter password: Sing
Invalid username or password. Exiting program.
```