### **Lab 7:** **Structures in Swift**

This lab will guide you through creating and manipulating structures in Swift. Follow the steps below to complete the assignment.

**Objectives:**

* Understand basic structure syntax and usage.
* Create and initialize structures with stored properties.
* Implement methods within structures.
* Use mutating methods to modify structure properties.
* Work with structures that contain computed properties.

**Instructions:**

**Step 1: Basic Structure Syntax and Usage**

* Define a simple structure called Book that has two properties: title and author.

struct Book {

var title: String

var author: String

}

// Using the Book structure

let book = Book(title: "To Kill a Mockingbird", author: "Harper Lee")

print("Title: \(book.title), Author: \(book.author)") // Output: Title: To Kill a Mockingbird, Author: Harper Lee

**Step 2: Initializing Structures with Stored Properties**

* Add a method to the Book structure to display the book's details.

struct Book {

var title: String

var author: String

func displayDetails() -> String {

return "Title: \(title), Author: \(author)"

}

}

// Using the displayDetails method

print(book.displayDetails()) // Output: Title: To Kill a Mockingbird, Author: Harper Lee

**Step 3: Implementing Methods within Structures**

Add a method to the Book structure to check if the book is written by a specific author.

struct Book {

var title: String

var author: String

func displayDetails() -> String {

return "Title: \(title), Author: \(author)"

}

func isWrittenBy(\_ authorName: String) -> Bool {

return author == authorName

}

}

// Using the isWrittenBy method

print(book.isWrittenBy("Harper Lee")) // Output: true

print(book.isWrittenBy("George Orwell")) // Output: false

**Step 4: Using Mutating Methods to Modify Structure Properties**

* Add a mutating method to the Book structure to update the title.

struct Book {

var title: String

var author: String

func displayDetails() -> String {

return "Title: \(title), Author: \(author)"

}

func isWrittenBy(\_ authorName: String) -> Bool {

return author == authorName

}

mutating func updateTitle(newTitle: String) {

title = newTitle

}

}

// Using the mutating method

var mutableBook = Book(title: "To Kill a Mockingbird", author: "Harper Lee")

mutableBook.updateTitle(newTitle: "Go Set a Watchman")

print(mutableBook.displayDetails()) // Output: Title: Go Set a Watchman, Author: Harper Lee

**Step 5: Working with Computed Properties**

* Add a computed property to the Book structure to return a description of the book.

struct Book {

var title: String

var author: String

func displayDetails() -> String {

return "Title: \(title), Author: \(author)"

}

func isWrittenBy(\_ authorName: String) -> Bool {

return author == authorName

}

mutating func updateTitle(newTitle: String) {

title = newTitle

}

var description: String {

return "The book '\(title)' is written by \(author)."

}

}

// Using the computed property

print(mutableBook.description) // Output: The book 'Go Set a Watchman' is written by Harper Lee.

**Conclusion**

In this lab, you learned how to:

* Define and use basic structures.
* Initialize structures with stored properties.
* Implement methods within structures.
* Use mutating methods to modify structure properties.
* Work with structures that contain computed properties.

Make sure to test your structures thoroughly and document your code with comments explaining each part of the logic. If you have any questions or need further clarification, feel free to ask. Happy coding!