BankAccount Class Assignment

Overview

Create a `BankAccount` class that simulates a simple bank account with basic deposit and withdrawal functionality.

Requirements

Class Structure

Your `BankAccount` class should have the following methods:

1. **`__init__()`**
    - Initialize a bank account with an initial balance
    - Handle the case where no initial balance is provided
    - Store the balance as an instance variable
2. **`deposit()`**
    - Deposit money into the account
    - Only allow positive amounts
    - Return the updated balance
    - If amount is not positive, don't change the balance
3. **`withdraw()`**
    - Withdraw money from the account
    - Only allow positive amounts
    - Return the updated balance if successful
    - Return "Insufficient Funds" if withdrawal exceeds balance
    - Don't change the balance if withdrawal exceeds available funds
4. **`get_balance()`**
    - Return the current balance of the account

Expected Behavior

Example Usage

```
# Create account with initial balance
account = BankAccount(100)
print(account.get_balance())  # Should output: 100

# Deposit money
account.deposit(50)
```

```
print(account.get_balance())  # Should output: 150

# Withdraw money (sufficient funds)
account.withdraw(30)
print(account.get_balance())  # Should output: 120

# Try to withdraw more than available
result = account.withdraw(200)
print(result)  # Should output: "Insufficient Funds"
print(account.get_balance())  # Should output: 120 (unchanged)
```

Test Cases

Your implementation should pass all the following test cases:

1. **Default initialization**: Account should handle creation without initial balance
2. **Custom initialization**: Account starts with given balance
3. **Single deposit**: Deposit increases balance correctly
4. **Multiple deposits**: Multiple deposits accumulate correctly
5. **Successful withdrawal**: Withdrawal decreases balance when funds are available
6. **Insufficient funds**: Withdrawal returns error message when funds are insufficient
7. **Exact balance withdrawal**: Can withdraw the exact balance amount
8. **Operation sequence**: Complex sequence of deposits and withdrawals works correctly

Implementation Tips

- Use instance variables to store the account balance
- Validate input amounts (only positive values)
- Handle edge cases like insufficient funds gracefully
- Return appropriate values from each method
- Make sure your class follows the exact method signatures shown above