1. **Write Java unit test cases or python unit test cases for all the requirements mentioned below. Submit the junit test case file or python test case file and I can execute the test cases.**

Program1:
Java program

```
import org.junit.Test;
import static org.junit.Assert.assertEquals;

public class BankAccount {
    private String accountNumber;
    private double balance;

    public BankAccount(String accountNumber, double initialBalance) {
        this.accountNumber = accountNumber;
        this.balance = initialBalance;
    }

    public String getAccountNumber() {
        return accountNumber;
    }

    public double getBalance() {
        return balance;
    }

    public void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
        }
    }

    public void withdraw(double amount) {
        if (amount > 0 && balance >= amount) {
            balance -= amount;
        }
    }
}

}
```

Python Program:

```
class BankAccount:
    def __init__(self, account_number, initial_balance):
        self.account_number = account_number
        self.balance = initial_balance

    def get_account_number(self):
```

```
        return self.account_number

    def get_balance(self):
        return self.balance

    def deposit(self, amount):
        if amount > 0:
            self.balance += amount

    def withdraw(self, amount):
        if amount > 0 and self.balance >= amount:
            self.balance -= amount
```

Requirements:

1. Account Creation:

- Requirement: The `BankAccount` class should allow creating an account with a specified account number and initial balance.

2. Account Details Retrieval:

- Requirement: The `getAccountNumber` method should return the account number associated with the account.
- Requirement: The `getBalance` method should return the current balance of the account.

3. Deposit Functionality:

- Requirement: The `deposit` method should increase the account balance by the deposited amount.
- Requirement: Depositing a negative amount should not modify the account balance.
- Requirement: Depositing zero should not modify the account balance.

4. Withdrawal Functionality:

- Requirement: The `withdraw` method should decrease the account balance by the withdrawn amount.
- Requirement: Withdrawing a negative amount should not modify the account balance.
- Requirement: Withdrawing an amount greater than the account balance should not modify the account balance.

5. Withdrawal Limits:

- Requirement: The `withdraw` method should only allow withdrawing an amount that is less than or equal to the current balance.

6. Account State After Transactions:

- Requirement: The account balance after depositing and withdrawing should accurately reflect the transactions performed.

7. Error Handling:

- Requirement: The account balance should not go below zero due to withdrawals.

Example:

```
 @Test
public void testDeposit()
 {
   BankAccount account = new BankAccount("12345", 100);
    account.deposit(50);
assertequals();
}
```