# CITP 190 – Introduction to Java
# Project 8

This project requires you to create one project. The project will work with customer bank accounts. Please be sure to download and unzip the project start files.

To receive full credit for this project you must submit the following:

- A detailed design diagram for each program.
- The source code for each program (the .java file) following the coding standards.
- The bytecode for each program (the.class file)
- Proof of the correctness of your output using the test data provided. Please note that you must provide proof for all test data. You may provide additional test data.
- A capture of the output of the program. The output must show all data from your proof.

Submit all files as one (1) ZIP file to the Project 8 Drop Box in the course site.

## Grading:

| | |
|---|---|
| Program design | 5 points |
| Design diagrams | 5 points |
| Following course standards | 5 points |
| Proof | 3 points |
| Screen captures | 2 points |

***Important notes:***

1. Incorrect calculations will result in a 0 grade for this project.
2. Output that is not presented as shown (including spaces and spelling) will result in a 0 grade for this project.
3. Code that does not follow the standards will result in a 0 grade for this project.
4. Uploading more than one zip file will result in a 0 grade for this project.
5. Not using all the test data provided will result in a 0 grade for this project.
6. The only change allowed to AccountApp.java is to add comments to the top of the file. The comments must include your name, the course code, the project number, and a statement that the code was given to you as part of the project. Any other changes will result in a 0 grade for this project. **It is very important to read through the AccountApp.java file to determine the names for methods in other classes. You may not change the code in the AccountApp.java file.**
7. You may change the MyValidator.java file. You must add comments at the tom that include your name, the course code, the project number, and a statement that the code was given to you as part of the project. You must clearly comment any other changes you make to the file. If you do not comment the changes you make to the file you will receive a 0 grade for this project.

# Project 8: Calculate the monthly balances of two bank accounts

## Console

```
Welcome to the Account application

Starting Balances
Checking: $1,000.00
Savings:  $1,000.00

Enter the transactions for the month

Withdrawal or deposit? (w/d): w
Checking or savings? (c/s): c
Amount?: 500

Continue? (y/n): y

Withdrawal or deposit? (w/d): d
Checking or savings? (c/s): s
Amount?: 200

Continue? (y/n): n

Monthly Payments and Fees
Checking fee:              $1.00
Savings interest payment:  $12.00

Final Balances
Checking: $499.00
Savings:  $1,212.00
```

## Operation

- The application begins by displaying the starting balances for a checking and savings account.
- The application prompts the user to enter the information for a transaction, including whether a withdrawal or deposit is to be made, whether the transaction will be posted to the checking or savings account, and the amount of the transaction.
- When the user finishes entering deposits and withdrawals, the application displays the fees and payments for the month followed by the final balances for the month.

## Specifications

- Create interfaces named `Depositable`, `Withdrawable`, and `Balanceable` that specify the methods that can be used to work with accounts. The `Depositable` interface should include this method:

  ```
  public void deposit(double amount)
  ```

  The `Withdrawable` interface should include this method:

  ```
  public void withdraw(double amount)
  ```

  And the `Balanceable` interface should include these methods:

  ```
  public double getBalance()
  public void setBalance(double amount)
  ```

- Create a class named `Account` that implements all three of these interfaces. This class should include an instance variable for the balance.
- Create a class named `CheckingAccount` that inherits the Account class. This class should include an instance variable for the monthly fee that's initialized to the value that's passed to the constructor. This class should also include methods that subtract the monthly fee from the account balance and return the monthly fee.
- Create a class named `SavingsAccount` that inherits the Account class. This class should include instance variables for the monthly interest rate and the monthly interest payment. The monthly interest rate should be initialized to the value that's passed to the constructor. The monthly interest payment should be calculated by a method that applies the payment to the account balance. This class should also include a method that returns the monthly interest payment.
- Create a class named `Transactions` that contains two static methods for depositing and withdrawing funds from either type of account:

```
public class Transactions
{
    public static void deposit(Depositable account,
    double amount)
    {
        account.deposit(amount);
    }

    public static void withdraw(Withdrawable account,
    double amount)
    {
        account.withdraw(amount);
    }
}
```

- Use the class named `AccountApp` to test your code. This is the class that contains the `main()` method for the program. The only change allowed to AccountApp.java is to add comments to the top of the file. The comments must include your name, the course code, the project number, and a statement that the code was given to you as part of the project.
- Use the `MyValidator` class or a variation of it to validate the user's entries. This validation code should not allow the user to withdraw more than the current account balance.

## Test Data

| Withdrawal or Deposit | Checking or Savings | Amount |
|---|---|---|
| W | C | $500.00 |
| D | S | $250.34 |
| W | C | $1245.34 |
| D | C | $1478.45 |
| W | S | $500.00 |