Inheritance

- A mechanism for enhancing existing classes. (advantage: code reuse)
- If two or more classes represent similar concepts or properties, then one class can **inherit** the properties of the other class.
- Example: SavingsAccount can inherit BankAccount
- Vocab:
- BankAccount is considered to be a "superclass" –> "parent class"
- SavingsAccount is "subclass" –> "child class"

BankAccount

```
public class BankAccount {
    private double balance;

    /**
    * Constructor for BankAccount
    * @param newBalance new Balance of account
    */
    public BankAccount (double newBalance){
        balance = newBalance;
    }

    /**
    * Deposits money into account
    * @param amount amount of money to deposit
    */
    public void deposit(double amount){
        balance += amount;
    }

    /**
    * Returns the balance of the account
    * @return balance of account
    */
    public double getBalance(){
        return balance;
    }
}
```

SavingsAccount

```
public class SavingsAccount extends BankAccount{
    private double interestRate;
```

```
    /**
     * Constructor for SavingsAccount
     * @param newBalance the new balance of the account
     */
    public SavingsAccount(double newBalance){
        super(newBalance);
        interestRate = 0.01;
    }

    /**
     * Adds new interest
     */
    public void addInterest(){
        double currentBalance = getBalance();
        deposit(balance * interestRate);
    }
}
```

- SavingsAccount automatically inherits all methods and instance variables of BankAccount.
- You can inherit all methods except for the constructor.
- SavingsAccount cannot access balance because it is not the owner of it.

Tester

```
public class Tester {
    public static void main (String [] args){
        SavingsAccount a = new SavingsAccount(1000);
        a.deposit(500);
        a.addInterest();
        System.out.println(a.getBalance()); // output 1515
    }
}
```