1. Log in
2. Unit 5
   - Inheritance (overriding and more)
   - Polymorphism
   - Abstract Classes
   - Interfaces

Oct 16-7:52 AM

Previous Lesson ...

*** Account Class**

- Review each item ...

```
public class Account {

    int number;
    String name;
    String password;

    public Account( ){
        number=0;
        name=" ";
        password="XYZ123";
    }
public void enterAccountInfo(int num, String nm, String pw){
        this.number = num;
        this.name=nm;
        this.password=pw;
    }
public int getAccountNumber(){
        return this.number;
    }
public String getPassword(){
        return this.password;
    }
public String getName(){
        return this.name;
    }

}
```

Nov 13-12:26 PM

Previous Lesson ...

*** Account Class**

- Review each item ...
- **NOTE**: getAccount Number( )

```
public class Account {

    int number;
    String name;
    String password;

    public Account( ){
        number=0;
        name=" ";
        password="XYZ123";
    }
public void enterAccountInfo(int num, String nm, String pw){
        this.number = num;
        this.name=nm;
        this.password=pw;
    }
public int getAccountNumber(){
        return this.number;
    }
public String getPassword(){
        return this.password;
    }
public String getName(){
        return this.name;
    }

}
```

Nov 13-12:26 PM

Previous Lesson ...

*** Account Class**

- Review each item ...
- **NOTE**: getAccount Number( )
  1. Don't need this unless a savings or checking account is created!

```
public class Account {

    int number;
    String name;
    String password;

    public Account( ){
        number=0;
        name=" ";
        password="XYZ123";
    }
public void enterAccountInfo(int num, String nm, String pw){
        this.number = num;
        this.name=nm;
        this.password=pw;
    }
public int getAccountNumber(){
        return this.number;
    }
public String getPassword(){
        return this.password;
    }
public String getName(){
        return this.name;
    }

}
```

Nov 13-12:26 PM

Previous Lesson ...

*** Account Class**

- Review each item ...
- **NOTE**: getAccount Number( )
  1. Don't need this unless a savings or checking account is created!
  2. The return statement is basically garbage since it gets overwritten (overriding).

```
public class Account {

    int number;
    String name;
    String password;

    public Account( ){
        number=0;
        name=" ";
        password="XYZ123";
    }
public void enterAccountInfo(int num, String nm, String pw){
        this.number = num;
        this.name=nm;
        this.password=pw;
    }
public int getAccountNumber(){
        return this.number;
    }
public String getPassword(){
        return this.password;
    }
public String getName(){
        return this.name;
    }

}
```

Nov 13-12:26 PM

Previous Lesson ...

*** Account Class**

- Review each item ...
- **NOTE**: getAccount Number( )
  1. Don't need this unless a savings or checking account is created!
  2. The return statement is basically garbage since it gets overwritten (overriding).
  3. We can simply remove the code and use the method as a "place-holder" to help us know future subclass methods!

```
public class Account {

    int number;
    String name;
    String password;

    public Account( ){
        number=0;
        name=" ";
        password="XYZ123";
    }
public void enterAccountInfo(int num, String nm, String pw){
        this.number = num;
        this.name=nm;
        this.password=pw;
    }
public int getAccountNumber(){

    }
public String getPassword(){
        return this.password;
    }
public String getName(){
        return this.name;
    }

}
```

Nov 13-12:26 PM

Previous Lesson ...

*** Account Class**

- Review each item ...
- **NOTE**: getAccount Number( )
  1. Don't need this!
  2. The return statement is garbage (overriding).
  3. Remove the code and use as a "place-holder"!
  4. This is called an Abstract Method (a method with no code that shows future implementation)

```
public class Account {

   int number;
   String name;
   String password;

   public Account( ){
      number=0;
      name=" ";
      password="XYZ123";
   }
public void enterAccountInfo(int num, String nm, String pw){
      this.number = num;
      this.name=nm;
      this.password=pw;
   }

public abstract int getAccountNumber( ){
   }

public String getPassword(){
      return this.password;
   }
public String getName(){
      return this.name;
   }
}
```

Nov 13-12:26 PM

Previous Lesson ...

*** Account Class**

- Review each item ...
- **NOTE**: getAccount Number( )
  1. Don't need this!
  2. The return statement is garbage (overriding).
  3. Remove the code and use as a "place-holder"!
  4. This is called an Abstract Method (a method with no code that shows future implementation)
  5. Use correct syntax! No brackets and semicolon!

* Must become an abstract class also ...

```
public abstract class Account {

   int number;
   String name;
   String password;

   public Account( ){
      number=0;
      name=" ";
      password="XYZ123";
   }
public void enterAccountInfo(int num, String nm, String pw){
      this.number = num;
      this.name=nm;
      this.password=pw;
   }

public abstract int getAccountNumber( );        //place-holder

public String getPassword(){
      return this.password;
   }
public String getName(){
      return this.name;
   }
}
```

Nov 13-12:26 PM

Previous Lesson ...

*** Account Class**

- Review each item ...
- **NOTE**: getAccount Number( )

* Now that it is abstract, it MUST be defined in a subclass!

```
public abstract class Account {

   int number;
   String name;
   String password;

   public Account( ){
      number=0;
      name=" ";
      password="XYZ123";
   }
public void enterAccountInfo(int num, String nm, String pw){
      this.number = num;
      this.name=nm;
      this.password=pw;
   }

public abstract int getAccountNumber( );        //place-holder

public String getPassword(){
      return this.password;
   }
public String getName(){
      return this.name;
   }
}
```

Nov 13-12:26 PM

Previous Lesson ...

*** Account Class**

- Review each item ...
- NOTE: getAccount Number( )
- NOTE: enterAccountInfo( )

```
public abstract class Account {

   int number;
   String name;
   String password;

   public Account( ){
      number=0;
      name=" ";
      password="XYZ123";
   }

public void enterAccountInfo(int num, String nm, String pw){
      this.number = num;
      this.name=nm;
      this.password=pw;
   }

public abstract int getAccountNumber( );        //place-holder

public String getPassword(){
      return this.password;
   }
public String getName(){
      return this.name;
   }
}
```

Nov 13-12:26 PM

Previous Lesson ...

*** Account Class**

- Review each item ...
- NOTE: getAccount Number( )
- NOTE: enterAccountInfo( )
  1. Not overriding, but overloading (Savings class used balance as a paramter too)

```
public abstract class Account {

   int number;
   String name;
   String password;

   public Account( ){
      number=0;
      name=" ";
      password="XYZ123";
      }

public void enterAccountInfo(int num, String nm, String pw){
      this.number = num;
      this.name=nm;
      this.password=pw;
   }

public abstract int getAccountNumber( );        //place-holder

public String getPassword(){
      return this.password;
   }
public String getName(){
      return this.name;
   }
}
```

Nov 13-12:26 PM

Previous Lesson ...

*** Account Class**

- Review each item ...
- NOTE: getAccount Number( )
- NOTE: enterAccountInfo( )
  1. Not overriding, but overloading (Savings class used balance as a paramter too)
  2. We probably won't ever access this since Account info should be entered upon creating a Savings and/or Checking

```
public abstract class Account {

   int number;
   String name;
   String password;

   public Account( ){
      number=0;
      name=" ";
      password="XYZ123";
   }

public void enterAccountInfo(int num, String nm, String pw){
      this.number = num;
      this.name=nm;
      this.password=pw;
   }

public abstract int getAccountNumber( );        //place-holder

public String getPassword(){
      return this.password;
   }
public String getName(){
      return this.name;
   }
}
```

Nov 13-12:26 PM

Previous Lesson ...

*** Account Class**

- Review each item ...
- NOTE: getAccount Number( )
- NOTE: enterAccountInfo( )
  1. Not overriding, but overloading (Savings class used balance as a paramter too)
  2. We probably won't ever access this since Account info should be entered upon creating a Savings and/or Checking
  3. This can be an abstract method also ... override

```
public abstract class Account {

   int number;
   String name;
   String password;

   public Account( ){                          parameters must
      number=0;                                match subclass
      name=" ";
      password="XYZ123";
   }

public abstract void enterAccountInfo(int num, String nm, String pw, double bal);

public abstract int getAccountNumber( );          //place-holder

public String getPassword(){
      return this.password;
   }
public String getName(){
      return this.name;
   }
}
```

Nov 13-12:26 PM

Previous Lesson ...

*** Account Class**

- Review each item ...
- NOTE: getAccount Number( )
- NOTE: enterAccountInfo( )

```
public abstract class Account {

   int number;
   String name;
   String password;

   public Account( ){
      number=0;
      name=" ";
      password="XYZ123";
   }

public abstract void enterAccountInfo(...parameters...);
public abstract int getAccountNumber( );

public String getPassword(){
      return this.password;
   }
public String getName(){
      return this.name;
   }
}
```

Nov 13-12:26 PM

Previous Lesson ...

*** Account Class**

- Review each item ...
- NOTE: getAccount Number( )
- NOTE: enterAccountInfo( )
- NOTE: earnInterest( )
  1. It is not in Account ...

```
public abstract class Account {

   int number;
   String name;
   String password;

   public Account( ){
      number=0;
      name=" ";
      password="XYZ123";
   }

public abstract void enterAccountInfo(...parameters...);
public abstract int getAccountNumber( );

public String getPassword(){
      return this.password;
   }
public String getName(){
      return this.name;
   }
}
```

Nov 13-12:26 PM

Previous Lesson ...

*** Account Class**

- Review each item ...
- NOTE: getAccount Number( )
- NOTE: enterAccountInfo( )
- NOTE: earnInterest( )
  1. It is not in Account ...
  2. But, all accounts use it ...

```
public abstract class Account {

   int number;
   String name;
   String password;

   public Account( ){
      number=0;
      name=" ";
      password="XYZ123";
   }

public abstract void enterAccountInfo(...parameters...);
public abstract int getAccountNumber( );

public String getPassword(){
      return this.password;
   }
public String getName(){
      return this.name;
   }
}
```

Nov 13-12:26 PM

Previous Lesson ...

*** Account Class**

- Review each item ...
- NOTE: getAccount Number( )
- NOTE: enterAccountInfo( )
- NOTE: earnInterest( )
  1. It is not in Account ...
  2. But, all accounts use it ...
  3. We should hold it's place

```
public abstract class Account {

   int number;
   String name;
   String password;

   public Account( ){
      number=0;
      name=" ";
      password="XYZ123";
   }

public abstract void enterAccountInfo(...parameters...);
public abstract int getAccountNumber( );

public abstract earnInterest( );

public String getPassword(){
      return this.password;
   }
public String getName(){
      return this.name;
   }
}
```

Nov 13-12:26 PM

Previous Lesson ...

*** Account Class**

- Review each item ...
- NOTE: getAccount Number( )
- NOTE: enterAccountInfo( )
- NOTE: earnInterest( )

```
public abstract class Account {

   int number;
   String name;
   String password;

   public Account( ){
      number=0;
      name=" ";
      password="XYZ123";
   }

public abstract void enterAccountInfo(...parameters...);
public abstract int getAccountNumber( );
public abstract void earnInterest( );

public String getPassword(){
      return this.password;
   }
public String getName(){
      return this.name;
   }
}
```

Nov 13-12:26 PM

Previous Lesson ...

*** Account Class**

- Review each item ...
- NOTE: getAccount Number( )
- NOTE: enterAccountInfo( )
- NOTE: earnInterest( )

* All abstract methods need to be defined in a Subclass or you will run into errors!

```
public abstract class Account {

    int number;
    String name;
    String password;

    public Account( ){
        number=0;
        name=" ";
        password="XYZ123";
    }

public abstract void enterAccountInfo(...parameters...);
public abstract int getAccountNumber( );
public abstract void earnInterest( );

public String getPassword(){
        return this.password;
    }
public String getName(){
        return this.name;
    }
}
```

Nov 13-12:26 PM

Abstract Classes ...

1. Typically is not an object itself, but a concept.
   Account is a concept ... checking/savings are objects.
2. Typically is never created, but subclasses are.
   Account isn't really created ... checking/savings are.
3. Do not technically have to have a constructor, subclass can.

Abstract Methods...

1. Methods with no code, just a place-holder to be over-written (overridden) in a subclass.
2. MUST be defined in a subclass or errors will occur.
3. Do not use braces (no code), simply ends with a semicolon.

Nov 18-10:22 AM

General Ideas behind Abstract Classes/Methods ...

```
public abstract class class1Name{

    /*  Instance Variables  */

    public abstract returnType method1Name( );
    public abstract returnType method2Name( );
    ...

    /*   Other Methods   */
}
```

```
public class class2Name extends class1Name{

    /*  Instance Variables  */

    /*  Constructor Method   */

    public returnType method1Name( ){
          /*   Code to implement method1   */
    }
    public abstract returnType method2Name( ){
          /*   Code to implement method2   */
    }

    /*   Other Methods   */
}
```

Nov 18-6:50 PM

Things to do ...

1. Wrap Up Unit 5 WS 01-04
2. Work on Unit 5 WS05 Abstract Classes

Oct 16-9:12 AM