Review first regarding Project 06

(1) Draw your own blueprint. (Write it in `House.java`)  
(2) (Program the computer to) make 1 (one) House following it. (Write the `HouseTesterApp.java` application.)

(1) and (2) are separate steps!
public class House {
    Picture favorite;
    Picture[] aOthers;
    public House() {
        this.aOthers = new Picture[3];
    }
    public void setFavoritePicture(Picture pRef) {
        this.favorite = pRef;
    }
    public void showFavoriteDOESNTCOUNT() {
        this.favorite.show();
    }
    public void showArtCollection() {
        ArtWall localWall = new ArtWall(600,500);
        //... You code!
        localWall.show();
    }
}
Blueprint for Houses

- Plaque for the address of your favorite picture
- Play at the order of an array of others
public class HouseTesterApp {
    public static void main(String[] a) {
        House myH = new House();
        FileChooser.pickMediaPath();
        myH.setFavoritePicture(
            new Picture(
                FileChooser.pickAPath()
            )
        );
        myH.showFavoriteDOESNTCOUNT();
    }
}
Main (...) {
    mH = new House();
}
public class House {
    Picture favorite;
    Picture[] aOthers;
    public House( )
    {
        this.aOthers = new Picture[3];
    }
    public void setFavoritePicture( Picture pRef )
    {
        this.favorite = pRef;
    }
    public void showFavoriteDOESNTCOUNT()
    {
        this.favorite.show( );
    }
    public void showArtCollection( )
    {
        ArtWall localWall = new ArtWall(600,500);
        //... You code!
        localWall.show();
    }
}
within a method that is called on an object, this refers to that object.
public class HouseTesterApp {
    public static void main(String[] a) {
        House myH = new House();
        FileChooser.pickMediaPath();
        String fName =
            FileChooser.pickAPPath();
        Picture testP = new Picture(fName);
        myH.setFavoritePicture(testP);
        myH.showFavoriteDOESNTCOUNT();
    }
}

(Electronic) Computers follow the instructions coded in their programs:

(A) Exactly in the order you have written them in the program, as modified by method calling, while/for loops, and if/if-else conditionals.

(B) In the order that makes sense to solve the problem correctly.

(C) Randomly
Course concept: Sequential execution.
What kinds of instructions can NOT you code in Java?

(A) Make a variable with a given name and type, like `int` or `double`.

(B) Copy a literal or computed value into a variable that had already been made.

(C) Print a warning when the program won't solve an assigned problem correctly.

(D) Print the value that ONCE WAS(in the past) the value of variable X, but was overwritten when the copy instruction `X = 98;` was performed.

(E) Neither (C) nor (D) can be coded in Java.
Course Concept: A variable (named storage location) and what you can program the computer to do with it.

Computer Science: Explore what computers CAN'T DO, together with what they can and how.
Make 2 variables with given names (technical word: declare)
int X;
int Y;

Copy 2 literal values into 2 variables the first times (tech: initialize)
X = 98;
Y = -1024;

Call twice the method that prints its parameter value on a separate line
System.out.println( X );
/* prints 98 */
System.out.println( Y );  //prints -1024

X = Y;
Y = X;
fails to swap the values!

System.out.println( X );  //NOW prints -1024, good.
System.out.println( Y );  //ALSO prints -1024, the swap failed!
Course concept: Preserving, moving around and otherwise using values of variables.

Skill: Tracing
System.out.println( X );
/* prints 98 */
System.out.println( Y ); //prints -1024

What's the best (college) answer?
(A) prints 98 and prints -1024 are examples of good, legal Java syntax, so 98 and -1024 are printed twice each.
(B) prints 98 and prints -1024 have Java syntax errors so the above 3 lines will not compile.
(C) prints 98 and prints -1024 are examples of good, legal Java syntax but 98 and -1024 are only printed once because the these print instructions are commented out.
(D) prints 98 and prints -1024 have Java syntax errors but it's OK because the /* */ and the // put them in comments.

Computer book style: Specific, syntactically correct code examples are set in typewriter font, including comments:
int X; /*Makes a variable and gives it the name X.*/

Text for people, not Java compilers, is set in Roman font like this.
/* x = x + 9; disable this operation for now*/

Commenting is also good for (temporarily) disabling code.

BUT: It's smart and wise to

1. Use some version snapshot saving software so you can always get a copy of your old work. (A cool modern one today is GIT, but there are many others)

2. Really remove all code that is wrong or has no logical purpose. (Because of 1., you won't worry!)
Course Concept: (Programming language) syntax.

Practice: Use and syntax of comments. Version history keeping.
Copy 2 literal values into 2 variables the first times (tech: initialize)
\[ x = 98; \]
\[ y = -1024; \]

Call twice the method that prints its parameter value on a separate line
\[
\text{System.out.println( } x \text{ );} \\
/* prints 98 */
\[
\text{System.out.println( } y \text{ );} \\
//prints -1024
\]

\[
\begin{align*}
\text{int } & \text{ temp; } & \text{succeeds!} \\
\text{x = y;} & \text{ YOU TRACE IT!} \\
\text{y = x;} & \text{(ask how)} \\
\text{temp = x;} & \\
\text{x = y;} & \\
\text{y = temp;}
\end{align*}
\]

fails to swap the values!

\[
\text{System.out.println( } x \text{ );} \\
//NOW prints -1024, good.
\]
\[
\text{System.out.println( } y \text{ );} \\
//ALSO prints -1024, the swap failed!
\]
Maybe we'll sketch problem solving sheets for swapping and rotating strategies, and scan them.

Concept: The swap pattern with a temporary variable.
College question: ASSUME `pIntArray` refers to a Java array full of non-negative ints. Write code to print the maximum value in that array.

```java
int maxSoFar;
maxSoFar = 0;
for( int i = 0; i < pIntArray.length; i++ )
{
    if( pIntArray[i] > maxSoFar )
    {
        maxSoFar = pIntArray[i];
    }
}
System.out.println( maxSoFar );
```
College question: **ASSUME** \( pA \) refers to a Java int array with non-zero length. Write code to print the maximum value in that array.

```java
int maxSoFar;
maxSoFar = pA[0];
for( int i = 1; i < pA.length; i++ )
{
    if( pA[i] > maxSoFar )
    {
        maxSoFar = pA[i];
    }
}
System.out.println( maxSoFar );
```
College question: ASSUME \textit{pA} refers to a Java int array with non-zero length. Would the alternative code make it find the maximum in all situations? (A) YES  (B) NO  (C) Maybe

```java
int maxSoFar;
maxSoFar = pA[0];
for( int i = 1; i < pA.length; i++ )
{
    if( pA[i] > maxSoFar )
        maxSoFar = pA[i];
}
System.out.println( maxSoFar );
```
```java
int maxSoFar;
maxSoFar = pA[0];
for( int i = 1; i < pIntArray.length; i++ )
{
    /* Alternative code */
    if( pA[i] > pA[i-1] )
    {
        maxSoFar = pA[i];
    }
}
System.out.println( maxSoFar );

May show on paper:  (A) Works for 1 2 3 4 3 2
(B) FAILS for 2 10 9 8 2 3
```
Concepts:

Array

Loop

For loop, generating the values 0, 1, 2, ..., length-1

Loop combined with an accumulator
College question: ASSUME \textit{pA} refers to a Java array of G&E \texttt{Picture} objects. Write code to print the maximum width of these \texttt{Pictures}.

```java
int maxSoFar;
maxSoFar = pA[0];
for( int i = 1; i < pA.length; i++ )
{
    if( pA[i] > maxSoFar )
    {
        maxSoFar = pA[i];
    }
}
System.out.println( maxSoFar );
```
What you might LOOK UP in G&E's book, documentation or code: The `getWidth()` method defined in the `Picture` class returns the width of the `Picture` as an `int`.

What you HAVE TO KNOW about Java:
(1) `Picture pA[];` declares variable pA so its value can refer to an array.

(2) `pA[ Some int numeric expression ]` denotes a `Picture` reference, JUST LIKE an “ordinary” variable declared like `Picture favorite;`

(3) Whatever you can do with `favorite`, like `favorite.getWidth( )`, you can do with `pA[ Some int numeric expression ]` like `pA[0].getWidth( )`
College question: ASSUME `pA` refers to a Java array of G&E `Picture` objects. Write code to print the maximum width of these `Pictures`.

```java
int maxSoFar;
maxSoFar = pA[0].getWidth();
for( int i = 1; i < pA.length; i++ )
{
    if( pA[i].getWidth() > maxSoFar )
    {
        maxSoFar = pA[i].getWidth();
    }
}
System.out.println( maxSoFar );
```
College question: ASSUME \texttt{pA} refers to a Java array of G&E \texttt{Picture} objects. Write code to print the sum of the widths of these \texttt{Pictures}...Even if somebody gave it an array of length 0!

```java
int sumOfWidths;
sumOfWidths = 0;
for( int i = 1; i < pA.length; i++ )
{
    sumOfWidths = pA[i].getWidth()+sumOfWidths;
    //The above line actually adds the width of
    //one more Picture into the sum.
}
System.out.println( sumOfWidths );
```