Useful computers are pre-programmed.
future diagram
2A Make a plan! Remember the goal. Pick, Position, Parametrize things you know might be pre-programmed steps.

2B Translate your positioned steps into Java code.

DO STUFF LIKE 2A ON PAPER scanned after class
REALLY DO THIS!!! It will make projects much much LESS FRUSTRATING!

Work out plans ON PAPER with PENCILS so you can erase and improve them!
public class GolfingTurtle extends Turtle
{
    public GolfingTurtle(World wref)
    {
        super( wref );
    }
    public void club( int sizeParamVar )
    {
        this.forward( sizeParamVar );
        //Purpose: Draw the club's handle.
        this.turn( 75 );
        //Purpose: Make the Turtle face in the head's direction.
        this.forward( sizeParamVar/10 );
        //Purpose: Draw the head.
        this.forward( - sizeParamVar/10 );
        //Purpose: Begin the retrace to bring the Turtle back to its starting state.
        this.turn( -75 );
        //Purpose: Undo the turn made before.
        this.forward( - sizeParamVar );
        //Purpose: Finish the retrace.
        return;
        //Purpose: Pre-program that the instructions will resume from where the method was called.
    }
    public static void main(String[] a)
    {
        System.out.println(“GolfingTurtle is NOT AN App!”);
    }
}
public class GolfClubDrawingApp
{
    public static void main(String[] a)
    {
        World wref = new World();
        GolfingTurtle tref = new GolfingTurtle( wref );
        tref.club( 200 ); // Draw a vertical club.
        tref.turn( 45 ); // Prepare to draw the 45 deg. club
        tref.club( 150 ); // Draw the 45 deg. one smaller.
        tref.turn( 90 ); // Prepare to draw a slant downward club.
        tref.club( 270 ); // Draw the down slanted club longest.
        tref.turn( -(45+90) ); // Restore Turtle's original state.
    }
}
What is this course about?

Serve students who want or need to actually program computers...

Designed for CS majors and minors...typically students headed for jobs or grad. studies (like CS Masters) where the expectation is “this candidate can program computers.”
Programming is like ice-skating, painting, playing sports or music, writing poetry and other creative hands-on endeavors in that it is a real misery and is impossible to do well if you need to but don't want to.
Specific facts and skills that will be graded

- How YOU, in ICSI201 Spr14 now, can pre-program a computer in Java (+bookClasses) AND ACTUALLY DEMONSTRATE your computer DOING what we specify.

- Understand, apply, and answer exam questions about the exact rules and vocabulary for “How ...” and plus what UNSEEN changes to stored data and other operations occur inside the computer.
Some specific professional “Hows” you must follow when YOU write Java in ICSI201

• Pre-program directions for DOING, in the FUTURE, big things inside DEFINITIONS OF YOUR OWN METHODS.
  – Pick, Place, Parametrize a sequence of
  – big (other methods calls) and
  – little things (Java primitive operations).

• Pre-program directions to ACTUALLY DO those big things by P.P.P. CALLS TO YOUR OWN METHODS inside YOUR Application's main method.
The Albany Way to pre-program your own methods (like `club()`) 

- (to start) extend G&E's `Turtle` class, like

```java
public class ArtisticTurtle extends Turtle {
    public ArtisticTurtle(World wref) {
        super(wref);
    }
    public void club(int sizeParam) {
        this.forward(sizeParam);
        /* ... */
        return;
        /* preprogrammed instr to draw one club*/
    }
}
```

- (don't program stuff into `Turtle.java`, like the book says.)
The Albany Way to USE your own pre-programmed methods (like club( ))

- USE your extended Turtle in main

```java
public class Proj3App {
    public static void main(String[] a) {
        World wref = new World();
        ArtisticTurtle tref = new ArtisticTurtle(wref);
        tref.club(100);
        tref.turn(55);
        tref.club(140);
    }
}
```

- Don't waste time typing Java commands again and again (like the book says.) Make an App, show it off, improve your safely saved code, TEST IT ONCE MORE, upload it for grade credit. Then celebrate.
Vocabulary word 1
Method DEFINITION (not call)

```java
public class ArtisticTurtle extends Turtle {

    public ArtisticTurtle(World wref) {
        super(wref);
    }

    public void club(int sizeParam) {
        this.forward(sizeParam);
        this.turn(75);
        this.forward(sizeParam/10);
        //... omitted stuff
        return;
    
    /*preprogrammed instr to draw one club
    and then return, ONLY WHEN the club
    method is called.*/
    }
}
```
public class Proj3App
{
    public static void main(String[] a)
    {
        World wref = new World();
        ArtisticTurtle tref = new ArtisticTurtle(wref);
        tref.club(100);
        /*Purpose: Actually draw the 1\textsuperscript{st} club.*/
        tref.turn(55);
        tref.club(140);
        /*Purpose: Draw the 2\textsuperscript{nd} club, bigger.*/
    }
}

A second, different CALL to the same method named \texttt{club} belonging to \texttt{ArtisticTurtles}. 
Live coding demo of starting Project 3!
(read the 4-page project assignment to review this and keep going)
Professional Practices taught in live code demo.

- Fresh, empty, separate directory/folder for each project.
- Incremental compiling and development—add just a little code and compile over and over.
- Indent to make the logical structure OBVIOUS.
- Document what a method is for and how SOMEBODY ELSE should use it in a Javadoc comment.