First name (color-in initial)	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	section (2,3,or 9)	first name initial	last name initial
Last name (color-in initial)	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z			

H10: Due Thu/Fri 02.09/02.10 in your assigned lab. Total Points: 50

MAY ONLY BE TURNED IN DURING THE CLASS INDICATED ABOVE, or offered in person, for in person grading, during instructor or TAs office hours. See the course syllabus at https://foo.cs.ucsb.edu/56wiki/index.php/W12:Syllabus for more details.

---

(1) (10 pts) Fill in the information below. Also, fill in the A-Z header by

- **coloring in** the first letter of your first and last name (as it would appears in Gauchospace),
- writing **either 2, 3 or 9** to indicate your **discussion section** meeting time
- writing your **first and last initial** in large capital letters.

All of this helps us to manage the avalanche of paper that results from the daily homework.

name:	
umail address:	@umail.ucsb.edu

Reading Assignment:

- Read just the first part of HFJ:Chapter_12, starting on p. **353** , and ending on p. 368.
- Earlier in the quarter we read the first part of HFJ:Chapter_17, pp. 581-595, about Jar files. Review those again.
- Now, we ready for the rest of the Chapter, i.e. pages **596** -606.
  - Consult the reading notes on the Wiki too---they are ESPECIALLY important when it comes to getting Java Web Start to work properly on CSIL.

You need this reading for lab04! (first half of Ch. 12, and 2nd half of Ch. 17)

- Ch. 12 is titled Getting GUI: A very graphic story, and we need these concepts for this week's lab:
  - How to create a JPanel 353-355.
  - How to draw 2-D graphics 363-368.
  - The material on pages 356-362 is about how to work with widgets, events, and ActionListeners, and we won't need that until lab05.
  - So, if you want, you can just skim over 356-368 for now, and come back to that when you do the reading for H11.
- The part of Ch. 17 that we skipped before is about Java Web Start (JWS)
  - JWS allows us to combine a JAR file, an XML file, and a web link so that we can load a Java application from the web and run it directly.
  - This is very handy when we want to run a GUI program in a convenient way.

---

(2) (4 pts) From Ch. 12, p. 353-355: What is the name of the Java class for an object that represents a window on the screen?

(3) (5 pts) From Chapter 17: Briefly, what is the purpose of Java Web Start? (I'm looking for a "big picture" answer here.)

Based on your reading in Chapter 12, p. 363-368:

(4) The text says: "If you want to put your own graphics on the screen, your best bet is to make your own paintable widget."

(a) (4 pts) The book recommends that to do this, you should: (fill in the blanks):

Make a _____ of JPanel,

and override one method: _____

(b) (3 pts) Now, several questions about this "mysterious method", i.e. the answer to the 2nd blank in part (a).

The book says that you write code to go inside this mysterious method when you override it. What kind of code do you write inside this mystery method? (Note: pp. 363-368 have lots of examples.)

(c) (3 pts) The book says that you never call this mysterious method yourself—and the reason is that the parameter to that method is something you don't have direct access to.

What is the Object that is the parameter to the mysterious method, that the System has access to, but you as a programmer don't?

(d) (3 pts) Though you can't call the mysterious method directly, there is a method that you **can** call that *asks* the system to call that method for you. What is this method that you **can** call?

(e) Inside this mysterious method, one often finds this mysterious line of code.

```
Graphics2D g2d = (Graphics2D) g;
```

- (3 pts) Explain WHAT this line of code is doing

- (5 pts) more importantly, Explain WHY that may be needed or helpful

(5) Back to Ch. 17: For JWS to work properly, three pieces of software on the client side must get involved: (1) the web browser, (2), the JWS Helper App, (3) the JVM on the client machine.

(a) (2 pts) Which of these requests the .jnlp file from the server?

(b) (2 pts) Which of these requests the .jar file from the server?

(c) (2 pts) Which of these runs the JAR file and executes the code inside?

(d) (2 pts) How does the component identified in part (c) know which class in the JAR to has the main() that should be executed?

(e) (2 pts) What is the language that the code inside the .jnlp file is written in?