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 (4, 5 or 6)	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			

H07: Due Thursday, 01.15 in Lab

Constructors, and Primitive Variables vs. Object References on the Stack and Heap (HFJ Ch9)
Assigned: Thu 01.08 Total Points: 50

MAY ONLY BE TURNED IN IN THE LECTURE/LAB LISTED ABOVE AS THE DUE DATE,
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/W15: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 appears in Gauchospace),
- writing **either 4, 5, or 6** to indicate your **discussion section (lab)** 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

If you collaborated with AT MOST one other person on this homework, write his/her name below. She/he should also have your name on his/her paper.

Reading Assignment:

In HFJ, Review Chapters 7 and 8, then read Chapter 9, which describes a **major difference** between C++ and Java: the issue of **garbage collection**. This is a *crucial* chapter, so read it *carefully*.

- HFJ, Chapter 7, **165** through 196 and reading notes HFJ:Chapter _7
- HFJ, Chapter 8, **197** through 235. HFJ:Chapter _8
- HFJ:Chapter_9, **235** Life and Death of an Object (Constructors)
- If there are reading notes on the wiki, consult those too—sometimes they contain helpful hints.

(2) (4 pts) Under what conditions does the compiler create a no-arg constructor for you?

(3) (4 pts) Under what conditions does the compiler NOT create a no-arg constructor for you?

(4) (16 pts) Given the following code excerpts:

```
public class Person {
   private String name;
   public Person (String name) {this.name = name;}
   public String getName() { return this.name;}
}
```

Write a class for Student that extends Person. Include a private attribute perm of type int. Include a constructor with the following signature:

```
public Student(String name, int perm) { ...
```

Use the proper technique (pp. 250-257) for invoking the parent class constructor (with a parameter) to initialize the name attribute.

(5) (16 pts) Based on what you learned from Chapter 9: Write a Java class that will compile and run (i.e. it needs a main() method) that has (ateast) the following four variables: a, b, c, and d, each instance of which will have the properties indicated. The class doesn't have to do any useful work---it is only to illustrate that you understand these concepts.

- a should be a primitive variable that will be stored on the stack
- b should be an object reference that will be stored on the stack (note: the references is on the stack, even though the object it refers to will always be on the Heap in Java.)
- c should be a primitive variable that will always be stored on the heap.
- d should be an object reference that will always be stored on the heap (note: here I want the reference variable itself to be on the heap, not just the object it refers to.)