Name: *(as it would appear on official course roster)*		
Umail address:	**@umail.ucsb.edu**	section 4, 5, or 6
Optional: name you wish to be called if different from name above.		
Optional: name of "homework buddy" (leaving this blank signifies "I worked alone"		

You may collaborate on this homework with AT MOST one person, an optional "homework buddy".

1

H00

CS56 W16

H00: Due Wednesday, 01.06 in Lecture

Quick look at Java Syntax and Objects (HFJ Ch1,2)
Assigned: Mon 01.04 Total Points: 50

MAY ONLY BE TURNED IN IN THE LECTURE/LAB LISTED ABOVE AS THE DUE DATE,
OR IF APPLICABLE, SUBMITTED ON GRADESCOPE. There is NO MAKEUP for missed assignments;
in place of that, we drop the five lowest scores (if you have zeros, those are the five lowest scores.)

---

1. (6 pts) Fill in the homework header properly—this helps us keep the grading pipeline flowing so that you get credit for your work and get feedback more quickly.
   - writing **either 4, 5, or 6** to indicate your **discussion section (lab)** meeting time
   - entering BOTH your name AND your umail address EVERY time.

   Paper submissions: One sheet of 8.5x11 paper double sided, or two DISCONNECTED SHEETS with your name on EACH. Please: **NO STAPLES, NO PAPERCLIPS, NO TAPE, NO ATTACHMENT OF ANY KIND**. These damage the document scanner.

   Scanned submission: When submitting by PDF upload: scan your pages legibly and **SCAN IN THE CORRECT ORDER**. Page 1 first, then Page 2, in the correct orientation. Failure to scan properly may result in zero credit, meaning you "use up" one of your five "drop the lowest grade" slots.

2. (5 pts) If you didn't do it yet: Login at https://github.ucsb.edu
   - Use your CSIL username/password to login.
   - Then you can immediately log out.

   This establishes your account on the local private UCSB Github server. Until you have logged in at least once, your login doesn't exist, so we can't add you to the CS56 "organization" and that is something we must do before Thursday's lab. **If it doesn't work**: Email help@engineering.ucsb.edu, and cc Prof. Conrad (pconrad@cs.ucsb.edu). Include "github.ucsb.edu access" in the subject line. Indicate whether the problem is CSIL access in general, or only access to github.ucsb.edu.

Write your CSIL username in the box here	

   Points will be awarded if you do this by the time this homework is graded. If not, you'll still have to do it, but you wont get those points back.

3. (5 pts) If you didn't do it yet: In addition to the UCSB campus github server, we will also be using the regular public github.com service for certain open source projects. So please:
   - create your github.com username and password (free account) at github.com
   - fill in the form at: http://bit.ly/cs56-w16-githubform

Write your github.com username in the box here	

   Points will be awarded if you do this by the time this homework is graded. If not, you'll still have to do it, but you wont get those points back.

READING ASSIGNMENT

Throughout the quarter, when I refer to **HFJ**, this means your Head First Java, 2nd Edition textbook---the one that has its own wiki page at HFJ.

- Please read HFJ:Chapter 1, and HFJ:Chapter 2
  - If you don't have your book yet, buy it! But in the meantime you can read this either at the UCSB library, or online.
  - Visit the course wiki at http://foo.cs.ucsb.edu/56wiki and you'll find the link to the online textbook.
- As you read, also consult the reading notes which you can find at this page on the wiki: HFJ:Chapter 1, HFJ:Chapter 2
- Then, do the problems on page 2 of this handout. Bring those with you to class to turn in on Wednesday.

4. (8 pts) On page 5 and 6, there is a set of exercises, and the answer to those. Here is a similar set of exercises, but the answer are not provided. Fill in the blanks.

java code	explanation
boolean cs56IsAwesome= true;	
Course c = new Course("CMPSC56","W16");	
String thisQuarter = "W16";	
if (c.getQuarter().equals(thisQuarter))	

5. (8 pts) Now, the same kind of exercise, but in reverse---I give you the description, you give me the code. These are designed that you should be able to just reason them out from the examples of Java code given on p.4 and p.5, and your general programming background from CS16 and CS24 in C/C++---no other knowledge of Java should be needed.

java code	explanation
	declare a variable that indicates whether this year is a leap year or not, and initialize it to say that it is not a leap year
	declare a variable item of type MenuItem, and initialize it to a "Caesar Salad" that costs 8.95. Assume those are the two parameters that the constructor takes.
	declare a variable of type double called total
	an assignment statement that calls the method getPrice on the variable item and adds the result into the variable total

6. (6 pts) According to chapter two, anytime you need to test a class in Java, you need at least two classes. (As we'll discuss in lecture, this isn't strictly true, but let's go with it for now.) In the (oversimplified) view the author is presented (it is just chapter two, after all,) what are the *two roles* that these two classes are playing? **Idenfity the roles, and explain each briefly,**

7. (6 pts) The author describes two different approaches to a problem—one taken by "Larry", and the other taken by "Brad". Based on what you read here, and in your own words, how would you characterize the main difference between the two approaches?

8. (6 pts) Throughout the book, some important material is sometimes in the little "side boxes" and dialogues. It's important to read everything on every page. One of these little side-boxes contains some important information about the "heap" in Java, and how it differs from the heap in C++. (C++ is not specifically mentioned, but from taking CS32, you should know about how the heap works in C++.) **Briefly explain** what the authors tell us about the heap in Java—something that is definitely different from the heap in C++.

Name: *(as it would appear on official course roster)*		
Umail address:	**@umail.ucsb.edu**	section 4, 5, or 6
Optional: name you wish to be called if different from name above.		
Optional: name of "homework buddy" (leaving this blank signifies "I worked alone"		

You may collaborate on this homework with AT MOST one person, an optional "homework buddy".

1

H01

CS56 W16

H01: Due Thursday, 01.07 in Lab

Variables, Types (double vs. float, primitive vs. reference etc.) Instance Variables, Methods (HFJ Ch3,4)
Assigned: Mon 01.04 Total Points: 46

MAY ONLY BE TURNED IN IN THE LECTURE/LAB LISTED ABOVE AS THE DUE DATE,
OR IF APPLICABLE, SUBMITTED ON GRADESCOPE. There is NO MAKEUP for missed assignments;
in place of that, we drop the five lowest scores (if you have zeros, those are the five lowest scores.)

Reading Assignment: Throughout the quarter, when I refer to **HFJ**, this means your Head First Java, 2nd Edition textbook.

- Read HFJ:Chapter_3 (especially pages 59-62) and reading notes on the wiki
- Read HFJ:Chapter_4 and reading notes on the wiki

Please:

- **No Staples**.
- **No Paperclips**.
- **No folded down corners**.

1. (6 pts) Fill in the homework header properly—this helps us keep the grading pipeline flowing so that you get credit for your work and get feedback more quickly.

    - writing **either 4, 5, or 6** to indicate your **discussion section (lab)** meeting time
    - entering BOTH your name AND your umail address EVERY time.

    Paper submissions: One sheet of 8.5x11 paper double sided, or two DISCONNECTED SHEETS with your name on EACH. Please: **NO STAPLES, NO PAPERCLIPS, NO TAPE, NO ATTACHMENT OF ANY KIND**. These damage the document scanner.

    Scanned submission: When submitting by PDF upload: scan your pages legibly and **SCAN IN THE CORRECT ORDER**. Page 1 first, then Page 2, in the correct orientation. Failure to scan properly may result in zero credit, meaning you "use up" one of your five "drop the lowest grade" slots.

2. Based on your reading in HFJ Chapter 3:

    a. (4 pts) If I write 3.4, is that of type double, or float?

    b. (4 pts) Declare x as a double and assign it the value 3.4 (as a double)

    c. (4 pts) Declare y as a float and assign it the value 3.4 (as a float)

3. (5 pts) In C++, the name of a plain old array of `Student` objects is not an object, but is rather a pointer to a `Student` (i.e. it is of type `Student *`. What about in Java—is an array an object, yes or no?

4. Variables that represent a primitive type (e.g. `boolean x;` or `int y;`) and variables containing object references (`String w;` or `Student z;`) have this in common—they are both composed of bits in memory. But—as explained in HFJ Chapter 3—they differ in what the bits *actually* represent. You won't get this one by just guessing—you really have to read the book.

   a. (4 pts) What do the bits that represent `int y;` represent?
   Assume that y is assigned the value 13

   b. (4 pts) What do the bits that represent `String w;` represent?
   Assume that w is assigned the value "foo".

5. Consider these questions about memory—answers are in Chapter 3 of HFJ.

   a. (2 pts) Does the amount of memory taken up by an object reference differ for different kinds of objects (say String vs. ArrayList<String>?)

   b. (2 pts) Does the amount of memory taken up by the object itself differ for different kinds of objects (assuming the same JVM)

   c. (2 pts) Can the amount of memory taking up for an object reference for a object particular type (say `String`) differ from one JVM to another?

6. Based on your reading in HFJ Chapter 3, p. 59-62 and HFJ Chapter 4 p. 84:

   a. (4 pts) Suppose I have a class called Student. How do I declare and allocate space for a plain old Java array called `students` that can hold 5 references to Student objects?

   b. (5 pts) Java `for` loops look pretty much just like C++ `for` loops (see HFJ page **10** if you really need to check.) Given that, assuming there is a default constructor `Student()` that you can call to create a new `Student` object, write a `for` loop that initializes all of the elements of the array `students` (from the previous problem) to be instances of the `Student` class.

Name: *(as it would appear on official course roster)*		
Umail address:	**@umail.ucsb.edu**	section 4, 5, or 6
Optional: name you wish to be called if different from name above.		
Optional: name of "homework buddy" (leaving this blank signifies "I worked alone"		

You may collaborate on this homework with AT MOST one person, an optional "homework buddy".

1

H02

CS56 W16

H02: Due Friday, 01.08 in GradeScope

Integer/String conversion, Random Numbers, Getters/Setters (HFJ Ch 4,5)
Assigned: Mon 01.04 Total Points: 50

MAY ONLY BE TURNED IN IN THE LECTURE/LAB LISTED ABOVE AS THE DUE DATE,
OR IF APPLICABLE, SUBMITTED ON GRADESCOPE. There is NO MAKEUP for missed assignments;
in place of that, we drop the five lowest scores (if you have zeros, those are the five lowest scores.)

- Review HFJ:Chapter_4 and reading notes
- Read Chapter 5 in HFJ, pages 95-124 and reading notes: HFJ:Chapter_5

For submission via Gradescope:

- Visit http://www.cs.ucsb.edu/~pconrad/gshints for hints on submission
- Scan pages in **correct order**.
- Email and paper submissions **NOT** accepted for GradeScope assignments.
- Start early. "I couldn't figure out GradeScope" is not an acceptable excuse.

1. (10 pts) Fill in the homework header properly—this helps us keep the grading pipeline flowing so that you get credit for your work and get feedback more quickly.

   - writing **either 4, 5, or 6** to indicate your **discussion section (lab)** meeting time
   - entering BOTH your name AND your umail address EVERY time.

   Paper submissions: One sheet of 8.5x11 paper double sided, or two DISCONNECTED SHEETS with your name on EACH. Please: **NO STAPLES, NO PAPERCLIPS, NO TAPE, NO ATTACHMENT OF ANY KIND**. These damage the document scanner.

   Scanned submission: When submitting by PDF upload: scan your pages legibly and **SCAN IN THE CORRECT ORDER**. Page 1 first, then Page 2, in the correct orientation. Failure to scan properly may result in zero credit, meaning you "use up" one of your five "drop the lowest grade" slots.

2. (10 pts) Write a few lines of code that demonstrate how to take a integer value that is in a String, and convert it to an integer value in an int variable. You can find an example of this in Chapter 5.

3. (10 pts) (From Chapter 5) Assume that `n` is an int variable that has already been assigned some value greater than or equal to 1. Write a few lines of Java code that declare a new int variable `x` and assign it a random integer between 0 and n-1 (inclusive, uniformly distributed over all n possible values.)

4. (10 pts) Frequently asked "job interview" question that comes from somewhere in chapter 4 or 5: briefly explain: part of Object-Oriented Programming is "encapsulation". What is "encapsulation"?

5. (10 pts) Based on your reading on p. 79 in Chapter 4: assume you have a class for a student with attributes name (of type String) and perm (of type int). Write setters and getters for name and perm as they would appear inside the student Class. The rest of the class has been written for you below---just fill in the missing parts. (Note that for purposes of this homework assignment, we have left out "public" and "private" since they are not yet covered in the book on p. 79, but later in the course you'd be expected to include them as appropriate.)

```
class Student {
  String name;
  int perm;

  // Now, you please fill in getters and setters for name and perm here.
  // That is all you need to write for this class.
  // For full credit, follow the naming conventions illustrated on p. 79.








}
```

Name: *(as it would appear on official course roster)*		
Umail address:	**@umail.ucsb.edu**	section 4, 5, or 6
Optional: name you wish to be called if different from name above.		
Optional: name of "homework buddy" (leaving this blank signifies "I worked alone"		

You may collaborate on this homework with AT MOST one person, an optional "homework buddy".

1

H03

CS56 W16

H03: Due Monday, 01.11 in Lecture

plain old Java arrays vs. ArrayList, initialization of instance variables, foreach style loop (HFJ 4,5,6)
Assigned: Wed 01.06 Total Points: 50

MAY ONLY BE TURNED IN IN THE LECTURE/LAB LISTED ABOVE AS THE DUE DATE,
OR IF APPLICABLE, SUBMITTED ON GRADESCOPE. There is NO MAKEUP for missed assignments;
in place of that, we drop the five lowest scores (if you have zeros, those are the five lowest scores.)

- Read HFJ:Chapter_6 pp. 125-164. Using the Java Library
- Some of these questions also come from HFJ chapters 4 and 5

Reminder:

- **Dark black pen** or **heavy dark pencil** scans best. Light pencil scans badly. If we can't read your answer, you might not get credit for it.
- **Short precise answers**. Don't Ramble.

1. (6 pts) Fill in the homework header properly—this helps us keep the grading pipeline flowing so that you get credit for your work and get feedback more quickly.
    - writing **either 4, 5, or 6** to indicate your **discussion section (lab)** meeting time
    - entering BOTH your name AND your umail address EVERY time.

   Paper submissions: One sheet of 8.5x11 paper double sided, or two DISCONNECTED SHEETS with your name on EACH. Please: **NO STAPLES, NO PAPERCLIPS, NO TAPE, NO ATTACHMENT OF ANY KIND**. These damage the document scanner.

   Scanned submission: When submitting by PDF upload: scan your pages legibly and **SCAN IN THE CORRECT ORDER**. Page 1 first, then Page 2, in the correct orientation. Failure to scan properly may result in zero credit, meaning you "use up" one of your five "drop the lowest grade" slots.

2. Review the difference between plain old java arrays (as in Chapter 4) and the ArrayList type (as in Chapter 6). Assume that a class called Student exists.
    a. (8 pts) write a line of java that makes a plain old Java array (Chapter 4 style) of Student references of size 5. (Don't allocate the Student objects, just the array of references, initially null).

    b. (8 pts) Now, write a line of java that makes an ArrayList of Student references (Chapter 6 style). Capacity is unimportant---choose 5, or take the default, whatever you like. (Don't allocate the Student objects, just the ArrayList, initially empty).

3. (8 pts) Based on your reading in HFJ Chapter 5: Java 1.5 introduced a new (to Java) kind of for loop sometimes called a "foreach" loop (even though foreach is not a keyword in Java)—your textbook calls it the "enhanced for loop". HFJ provides an overview of this kind of loop on p. 105 and 116. Write a few lines of code that declare an array of five integers, initializing them to the first five prime numbers (you can use a literal array initializer here—you don't need to write code to compute the prime numbers), and then write a foreach type loop that iterates through that array printing out the values, one on each line.

4. (10 pts) Based on your reading in HFJ Chapter 4, consider the Java code in the box at right, and these three questions in the space provided. For full credit, answer part (a), then answer parts (b) and (c) or leave them blank as appropriate.

Contents of `TestCode.java`

```
public class TestCode {
    public static void main (String[] args) {
        String name;
        int perm;
        System.out.println("Student's perm is " + perm) ;
        System.out.println("Student's name is " + name ) ;
    }
}
```

   a. Will this code produce an error message, when compiled with `javac TestCode.java` and if so what? (I don't need a detailed character by character account of the error messsage—just a general description of what the error is will be sufficient.)

   b. IF IT DOES COMPILE: will this code produce an error message, when run with `java TestCode` and if so what? (same as the previous question—just a general description of the error is sufficient.) (It it doesn't compile, leave this and the next question blank.)

   c. IF IT DOES COMPILE AND RUN WITHOUT AN ERROR, what will be the resulting output when this code is run? (If it will not run without error, leave this question blank.)

SPACE FOR answer for 4a, 4b, 4c:

5. (10 pts) Based on your reading in HFJ Chapter 4, consider the Java code in the box at right, and then answer the three questions in the space provided. For full credit, answer part (a), then answer parts (b) and (c) or leave them blank as appropriate.

   a. Will this code produce an error message, when compiled with `javac *.java` and if so what? (I don't need a detailed character by character account of the error messsage—just a general description of what the error is will be sufficient.)

   b. IF IT DOES COMPILE: will this code produce an error message, when run with `java StudentTestDrive` and if so what? (same as the previous question—just a general description of the error is sufficient.) (It it doesn't compile, leave this and the next question blank.)

   c. IF IT DOES COMPILE AND RUN WITHOUT AN ERROR, what will be the resulting output when this code is run? (If it will not run without error, leave this question blank.)

Contents of `Student.java`

```
class Student {
    private int perm;
    private String name;

    public int getPerm() {
        return perm;
    }
    public String getName() {
        return name;
    }
}
```

Contents of `StudentTestDrive.java`

```
public class StudentTestDrive {
    public static void main (String[] args) {
        Student s = new Student() ;
        System.out.println("Student's perm is " + s.getPerm() ) ;
        System.out.println("Student's name is " + s.getName() ) ;
    }
}
```

SPACE FOR answer for 5a, 5b, 5c:

Name: *(as it would appear on official course roster)*		
Umail address:	**@umail.ucsb.edu**	section 4, 5, or 6
Optional: name you wish to be called if different from name above.		
Optional: name of "homework buddy" (leaving this blank signifies "I worked alone"		

You may collaborate on this homework with AT MOST one person, an optional "homework buddy".

1

H04

CS56 W16

H04: Due Tuesday, 01.12 in GradeScope

Inheritance, Interfaces, Abstract Classes, Polymorphism (HFJ Ch 7,8)
Assigned: Wed 01.06 Total Points: 50

MAY ONLY BE TURNED IN IN THE LECTURE/LAB LISTED ABOVE AS THE DUE DATE,
OR IF APPLICABLE, SUBMITTED ON GRADESCOPE. There is NO MAKEUP for missed assignments;
in place of that, we drop the five lowest scores (if you have zeros, those are the five lowest scores.)

Reading Assignment:

We are moving right along, reading two more chapters. These two chapters are short, and a good bit of this is basic review of OOP concepts you may have already seen in CS24 (and possibly in CS32 if you took that, which is recommended, though not required, as a pre-req to CS56). The coverage of these chapters will be limited on the first midterm exam, since the programming assignments that reinforce this material aren't due until after the exam. But you need to read this material now and get comfortable with the ideas before you tackle the first lab that uses inheritance. We also need at least a basic understanding of inheritance and polymorphism before we can do assignments involving GUIs and graphics, or tackle the open source projects.

For submission via Gradescope:

- Visit http://www.cs.ucsb.edu/~pconrad/gshints for hints on submission
- Scan pages in **correct order**.
- Email and paper submissions **NOT** accepted for GradeScope assignments.
- Start early. "I couldn't figure out GradeScope" is not an acceptable excuse.

- HFJ:Chapter_7, **165** through 196, and reading notes
- HFJ:Chapter_8, **197** through 235, and reading notes
- Also: Review HFJ:Chapter_5 and HFJ:Chapter_6 and reading notes.

1. (6 pts) Fill in the homework header properly—this helps us keep the grading pipeline flowing so that you get credit for your work and get feedback more quickly.

   - writing **either 4, 5, or 6** to indicate your **discussion section (lab)** meeting time
   - entering BOTH your name AND your umail address EVERY time.

   Paper submissions: One sheet of 8.5x11 paper double sided, or two DISCONNECTED SHEETS with your name on EACH. Please: **NO STAPLES, NO PAPERCLIPS, NO TAPE, NO ATTACHMENT OF ANY KIND**. These damage the document scanner.

   Scanned submission: When submitting by PDF upload: scan your pages legibly and **SCAN IN THE CORRECT ORDER**. Page 1 first, then Page 2, in the correct orientation. Failure to scan properly may result in zero credit, meaning you "use up" one of your five "drop the lowest grade" slots.

2. (12 pts) Based on your reading in HFJ:Chapter_7:

   Complete the following exercise from p. 179, putting a check next to the relationships that make sense.

	Oven extends Kitchen
	Guitar extends Instrument
	Person extends Employee
	Ferrari extends Engine
	FriedEgg extends Food

	Beagle extends Pet
	Container extends Jar
	Metal extends Titanium
	GratefulDead extends Band
	Blonde extends Smart
	Beverage extends Martini

3. (8 pts) Based on your reading in HFJ:Chapter_7: What does it mean to have a "polymorphic argument" or a "polymorphic return type" for a method? Explain with an example—but NOT using the example of Vets and Animals used in the book. Substitute your own example. Give a detailed enough description of the class hierarchy you have in mind to make it clear that you get the concept.

4. (8 pts) Based on your reading in HFJ:Chapter_8: Briefly describe the difference between an abstract class and an interface.

5. (8 pts) What is one advantage of using an ArrayList over a plain old Java array?

6. (8 pts) Why do some classes in the Java API have package names that start with java.blah while others have package names that start with javax.blah? What does the x mean?

Name: *(as it would appear on official course roster)*		
Umail address:	**@umail.ucsb.edu**	section 4, 5, or 6
Optional: name you wish to be called if different from name above.		
Optional: name of "homework buddy" (leaving this blank signifies "I worked alone"		

You may collaborate on this homework with AT MOST one person, an optional "homework buddy".

1

H05

CS56 W16

H05: Due Wednesday, 01.13 in Lecture

Constructors, and Primitive Variables vs. Object References on the Stack and Heap (HFJ Ch9)
Assigned: Wed 01.06 Total Points: 50

MAY ONLY BE TURNED IN IN THE LECTURE/LAB LISTED ABOVE AS THE DUE DATE,
OR IF APPLICABLE, SUBMITTED ON GRADESCOPE. There is NO MAKEUP for missed assignments;
in place of that, we drop the five lowest scores (if you have zeros, those are the five lowest scores.)

Reading Assignment:

In HFJ, Review HFJ:Chapter _7 and HFJ:Chapter _8 then read HFJ: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*. If there are reading notes on the wiki, consult those too—sometimes they contain helpful hints.

1. (6 pts) Fill in the homework header properly—this helps us keep the grading pipeline flowing so that you get credit for your work and get feedback more quickly.
   - writing **either 4, 5, or 6** to indicate your **discussion section (lab)** meeting time
   - entering BOTH your name AND your umail address EVERY time.

   Paper submissions: One sheet of 8.5x11 paper double sided, or two DISCONNECTED SHEETS with your name on EACH. Please: **NO STAPLES, NO PAPERCLIPS, NO TAPE, NO ATTACHMENT OF ANY KIND**. These damage the document scanner.

   Scanned submission: When submitting by PDF upload: scan your pages legibly and **SCAN IN THE CORRECT ORDER**. Page 1 first, then Page 2, in the correct orientation. Failure to scan properly may result in zero credit, meaning you "use up" one of your five "drop the lowest grade" slots.

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. 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 (at least) 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.

- (5 pts) a should be a primitive variable that will be stored on the stack
- (5 pts) 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.)
- (5 pts) c should be a primitive variable that will always be stored on the heap.
- (5 pts) 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.)

Name: *(as it would appear on official course roster)*		
Umail address:	**@umail.ucsb.edu**	section 4, 5, or 6
Optional: name you wish to be called if different from name above.		
Optional: name of "homework buddy" (leaving this blank signifies "I worked alone"		

1

H06

CS56 W16

You may collaborate on this homework with AT MOST one person, an optional "homework buddy".

H06: Due Thursday, 01.14 in Lab

Static vs. Non-Static methods, Autoboxing, Auto-unboxing, Final, Math class (HFJ Ch10)
Assigned: Thu 01.07 Total Points: 50

MAY ONLY BE TURNED IN IN THE LECTURE/LAB LISTED ABOVE AS THE DUE DATE,
OR IF APPLICABLE, SUBMITTED ON GRADESCOPE. There is NO MAKEUP for missed assignments;
in place of that, we drop the five lowest scores (if you have zeros, those are the five lowest scores.)

Reading Assignment: : HFJ:Chapter_10, plus reading notes on the wiki.

1. (5 pts) Fill in the homework header properly—this helps us keep the grading pipeline flowing so that you get credit for your work and get feedback more quickly.
   - writing **either 4, 5, or 6** to indicate your **discussion section (lab)** meeting time
   - entering BOTH your name AND your umail address EVERY time.

   Paper submissions: One sheet of 8.5x11 paper double sided, or two DISCONNECTED SHEETS with your name on EACH. Please: **NO STAPLES, NO PAPERCLIPS, NO TAPE, NO ATTACHMENT OF ANY KIND**. These damage the document scanner.

   Scanned submission: When submitting by PDF upload: scan your pages legibly and **SCAN IN THE CORRECT ORDER**. Page 1 first, then Page 2, in the correct orientation. Failure to scan properly may result in zero credit, meaning you "use up" one of your five "drop the lowest grade" slots.

2. (3 pts) What does final mean when applied to a static variable?

3. (3 pts) What does final mean when applied to a non-static variable?

4. (3 pts) What does final mean when applied to a method?

5. (3 pts) What does final mean when applied to a class?

6. Be sure you understand *precisely* the difference between auto boxing and auto unboxing before answering the following questions.
   a. (4 pts) Write a line (or two) of Java code that would result in auto-unboxing but NOT auto-boxing

   b. (4 pts) Write a line (or two) of Java code that would result in boxing but NOT auto-unboxing

7. Answer these questions about static vs. non-static methods. Note: When I ask for for a "use case" for something, I mean "describe the general set of circumstances when such a thing is useful." It is not sufficient in these cases to just cite a single specific instance. For example, don't say "the Math class is an example of such and such" and leave it at that. Say something like "the Math class is an example, because ... " and then describe what is true about the Math class that, if true of another class Foo, would also make it a good candidate for this situation.

   a. (3 pts) What is a "use case" for a class with all static methods? (See note above about "use case").

      a. (2 pts) In this use case (the one you just described), does it make sense to create an instance this class? Why or why not? (briefly explain).

   b. (3 pts) What is a "use case" for a class with all non-static methods?

      a. (2 pts) In this use case (the one you just described), does it make sense to create an instance this class? Why or why not? (briefly explain).

   c. (3 pts) What is a "use case" for a class with a mix of static and non-static methods?

      a. (2 pts) In this use case (the one you just described), does it make sense to create an instance this class? Why or why not? (briefly explain).

8. (10 pts) Write a class in Java with a main program equivalent to the following C program. (Hint: Use the Math and classes and formatting methods of the String class described in Chapter 10.)

```
#include <math.h>
#include <stdio.h>

int main()
{
 int i;
 for (i=0;i<=10;i++)
 {
    double value= sin((i/10.0) * (2 * M_PI));
    printf("i=%3d  value=%6.3lf\n", i, value);
 }
 return 0;
}
```

Name: *(as it would appear on official course roster)*		
Umail address:	**@umail.ucsb.edu**	section 4, 5, or 6
Optional: name you wish to be called if different from name above.		
Optional: name of "homework buddy" (leaving this blank signifies "I worked alone"		

You may collaborate on this homework with AT MOST one person, an optional "homework buddy".

1

H07

CS56 W16

H07: Due Friday, 01.15 in GradeScope

Exceptions (HFJ Ch11)
Assigned: Thu 01.07 Total Points: 50

MAY ONLY BE TURNED IN IN THE LECTURE/LAB LISTED ABOVE AS THE DUE DATE,
OR IF APPLICABLE, SUBMITTED ON GRADESCOPE. There is NO MAKEUP for missed assignments;
in place of that, we drop the five lowest scores (if you have zeros, those are the five lowest scores.)

Reading Assignment:

- HFJ:Chapter_11 plus reading notes.

Note: The following point has been a frequent source of confusion in previous quarters of CS56.

Be sure you understand how each of these looks different in the code:

- Creating a new type of exception (involves creating a new class)
- Signalling that an exception has happened (constructing an instance of an Exception class and throwing it)
- Detecting that an exception has occurred (try/catch block)

For submission via Gradescope:

- Visit http://www.cs.ucsb.edu/~pconrad/gshints for hints on submission
- Scan pages in **correct order**.
- Email and paper submissions **NOT** accepted for GradeScope assignments.
- Start early. "I couldn't figure out GradeScope" is not an acceptable excuse.

1. (6 pts) Fill in the homework header properly—this helps us keep the grading pipeline flowing so that you get credit for your work and get feedback more quickly.

   - writing **either 4, 5, or 6** to indicate your **discussion section (lab)** meeting time
   - entering BOTH your name AND your umail address EVERY time.

   Paper submissions: One sheet of 8.5x11 paper double sided, or two DISCONNECTED SHEETS with your name on EACH. Please: **NO STAPLES, NO PAPERCLIPS, NO TAPE, NO ATTACHMENT OF ANY KIND**. These damage the document scanner.

   Scanned submission: When submitting by PDF upload: scan your pages legibly and **SCAN IN THE CORRECT ORDER**. Page 1 first, then Page 2, in the correct orientation. Failure to scan properly may result in zero credit, meaning you "use up" one of your five "drop the lowest grade" slots.

2. Exceptions in Java can be divided into two broad categories:

   - One category is the kind that, if there is any chance it can happen in the code has to be "caught or declared to be thrown"
   - The other category is the kind that can happen, but doesn't have to be "caught or declared to be thrown".

   a. (6 pts) Exceptions that do not have to be caught or declared to be thrown are extensions of (i.e. they are subclasses of) what class?

   b. (6 pts) What is the rationale for having some exceptions that do NOT have to be declared to be thrown or caught? (i.e. why did the designers of Java put that feature into the language?)

   c. (6 pts) What is the rationale for having some exceptions that DO have to be declared to be thrown or caught? (i.e. why did the designers of Java put that feature into the language?)

3. (10 pts) In the code excerpt below, there is a test to see if the course is in the list of courses. If it is not, we want to signal that an exception has happened. Assuming there is an exception called `NoSuchUCSBCourseException` (i.e. that class has already been written and exists), fill in the blank with a line of Java code that indicates that this exception has happened, and the program needs to signal that.

```
public Course lookupCourse(String courseNum) throws NoSuchUCSBCourseException {
    Hashtable<String,Course> ucsbCourses = getCoursesHashTable();
    Course result = ucsbCourses.get(courseNum);
    if (result == null) {
            // your answer goes here!








    }
    return result;
}
```

4. (10 pts) Assume that s is an object of type `Student`, and that there is a method `public void registerFor(String courseNum)` that might throw the `NoSuchUCSBCourseException`. Write a segment of Java code that will:

   - call `s.registerFor(someCourse);`
   - writes `"Success"` on `System.out` if the registration succeeded (i.e. that exception doesn't happen)
   - will write `"Sorry " + someCourse + "does not exist"` to `System.out` if that exception occurs.

5. (6 pts) Write the code that creates a new kind of exception called `BadSuitException`. This exception might be thrown, for example, by a program that expects a character that is one of 'H','D','C','S' for Hearts, Diamonds, Clubs and Spades when it encounters an illegal value (one other than 'H', 'D','C' or 'S'.) `BadSuitException` should be a subclass of `IllegalArgumentException`. Write ONLY the code that creates the new kind of exception.