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

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

H07: Due Friday, 01.15 in GradeScope

H07

CS56 W16

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.