CSE-11 Programming Assignment 3

Due Date: Wednesday, April 21, 11:59PM Pacific Time

Learning Goals

- Practice writing classes and conditionals
- Do some open-ended exploration of Java features

Collaboration

Different assignments in this course have different collaboration policies. On this assignment, you can collaborate with anyone in the course, including sharing code. In your submission, give credit to all students and course staff who helped you with this assignment by noting their name and how you used their ideas or work. Note that using someone's work without giving credit to them is a violation of academic integrity.

We've added a file called `CREDITS.txt` for you to write these credits into.

Resubmission/Late Policy

- We will not accept this PA late.
- If you did not submit this PA or did poorly (<75%), you can fix your PA and resubmit it to the PA resubsmission assignment in Gradescope to earn a maximum of 75% the points.
- If you scored higher than 75% on the origianl PA, we may grade the resubmission, but will not change your original grade.
- The resubmission will be open for 2 weeks after the PA's original due date.

Drill 1 (autograded)

Write three classes as described below. They do not need to have any methods or constructors. Save them in the file `Drill1.java`.

- A class named `A` with two fields, `int f1` and `String f2`
- A class named `B` with two fields, `A field1` and `String field2`
- A class named `C` with three fields, `B fieldB`, `A fieldA`, `int field3`

./run x

Drill 2 (autograded)

In the file `Drill2.java`, add the following class definitions (you can copy/paste them from here):

```
class C1 {
  C2 other;
  C1(C2 other) {
    this.other = other;
  }
}

class C2 {
  int x;
  C2(int x) {
    this.x = x;
  }
}
```

C2 object

constructor for C1

C2 myC2 = new C2(5);

myC2 is a reference to a C2 object

5, 10, 11

constructor for C2

Then add a class definition called `Drill2` with the following fields:

int field = 1;

C2 first = new C2(10);

- A field named `first` of type `C2` with its `x` field equal to 10
- A field named `second` of type `C1`. It's value should be a reference to a `C1` object with its `other` field set to any `C2` object other than the one stored in `first` (you can create another `C2` object for this).
- A field named `third` of type `C1`. It's value should be a reference to a `C1` object with its `other` field the same `C2` object as the one stored in the `first` field.

Use the constructor for C1

create a C1 object

Use the constructor

new C1(new C2(5))

create a C2 object

Drill 3 (autograded)

In the file `Drill3.java`, write a class called `TextTweet` that has two fields: one field called `contents` of type `String`, and one field called `likes` of type `int`. Give it a constructor of two arguments that initializes those fields. In it, write the following methods:

- `hasMention` which takes a `String` called `username` and checks if the string `@` followed by that username apppears in the Tweet contents, returning `true` if it does and `false` otherwise. (There are some interesting cases for this method. For example, to check if we have the username "dummy" in "@dummy1 @dummy2", hasMention() should return false in this case because username dummy1 and dummy2 is not the same as dummy, while "hello @dummy world" and "CSE 11 is a cool class @dummy" should return true)

@dummy is contents?

boolean hasMention(String username) {

...

}

- `hasLike` which takes no arguments and returns `true` if the tweet one or more likes, `false` otherwise.
- `firstMention` which takes no arguments and returns a `String` containing the substring between the first appearance of the `@` character in the `contents` and the first space character after that. Do not include the `@` character and the space character. If there is no space after the `@`, or if there's no `@`, the empty string `""` should be returned.

(You may find some String methods in the Java documentation useful)

Also in `Drill3.java`, write a class called `ReplyTweet` that has three fields: one called `replyTo` of type `TextTweet`, one called `contents` of type `String`, and one called `likes` of type `int`. Give it a constructor of three arguments that initializes these fields. In it, write the following methods:

- `morePopularReply` which takes no arguments and returns true if this ReplyTweet has more likes than the TextTweet it is replying to
- `allLikes` which takes no arguments and returns the sum of the likes on this ReplyTweet and on the TextTweet it is replying to
- `hasMention` which takes a `String` called `username` and checks if the string `@` followed by that username apppears in this `ReplyTweet`'s contents or in the `TextTweet` that is being replied to.

We highly recommend adding a class to this file called `Drill3` that has any tests or examples you used to help verify that your methods worked as you expected.

Drill 4 (autograded)

Math library

In a file called `Drill4.java`, write a class `Drill4` with the following methods (you may find some Math methods in the Java documentation useful):

- `phaseOfWater` which takes an `int` and returns `"vapor"` if the number is greater than or equal to 100, `"liquid"` if the number is less than 100 and greater than 0, and `"solid"` if the number is less than or equal to 0.
- `maxDifference` which takes three `ints` and returns the largest absolute difference between any two of them. For example, `maxDifference` applied to 1, -1, and 5 should return 6 because the difference between -1 and 5 is 6 which is greater than 2 or 4 (the differences between -1 and 1, and between 1 and 5).
- `ringArea` which takes two `double`s representing the radius of an inner circle and an outer circle, and returns the area of the ring between them as a `double`. Assume both inputs are positive and that the first number is smaller than the second. Recall that the area of a circle is πr^2. You can use `Math.PI`, a field conveniently defined for us by Java, as a constant for the value of π.

In addition, you must write at least three interesting tests for **each** method above, and include the expected values. You can test them in the same way as we did in class and previous two PAs. You can use the tester package if you want, but we're not grading for it. In PA4 we will explicitly require it in some cases to make sure everyone gets practice with it.

Open-Ended 1 (manually graded)

Consider the following statements about Java programs:

- **Statement A**: In Java, two different classes can define a field with the same name and type.
- **Statement B**: In Java, one class can define two fields with the same name as long as they have different types.
- **Statement C**: In Java, two different methods in the same class can have the same parameter name, and arguments passed to one of those methods don't affect the parameter in the other.

For each statement, write a small Java program that demonstrates whether it is true or false. In the programs you should define some classes that reflect the statements. You can choose whatever names you want for the classes. Put the class definitions in the files `Open1A.java`, `Open1B.java`, and `Open1C.java`. Inside the files, you should also include an `Open1A` / `Open1B` / `Open1C` class with some instances of the classes you just create.

To show that a statement is true, write a Java program that matches the statement, doesn't produce an error when run, and produces some meaningful output when `./run`.

To show that a statement is false, write a Java program that matches the statement and produces an error when run, demonstrating that Java programs cannot do what the property says.

Include both the program and the output of running the program without error as your submission; you can upload screenshots as images clearly named with `Open1A`, `Open1B`, etc in the title, or copy-paste the text output into a comment in your code.

Open-Ended 2 (manually graded)

Create a class named `R` that has a field of type `String` and a field of type `R`. Give it a constructor that initializes both fields. Put the class in a file called `ExamplesR.java`. Add an `ExamplesR` class to this file, and answer the following questions in that file:

1. Construct an example `R` object. Were you able to? Explain your example if you were able to, and explain why you think it's not possible if you weren't.
2. On Twitter, it's possible to reply to a reply to a Tweet (that's not a typo, it's a reply to a reply). This is true of many systems, like email, Facebook comments, Piazza followups, and so on. With the class structure in `Drill3` with `ReplyTweet` and `TextTweet` (that is, without changing the fields as described above), could you construct an example of a reply to a reply to a Tweet? Why or why not?

Submission

You will submit all of your files to the `pa3` assignment on Gradescope:

- On the Gradescope upload screen, you can keep clicking "Browse Files" to select more than one file for your upload; you can select them one at a time or use your operating system's multi-select (Shift-Click usually works) to select them all and drag them onto the upload area, or other options that you find that work.
- You can also make a zip archive of all of your files and upload them all at once if you prefer.

The parts marked `autograded` will be graded automatically in Gradescope based on tests we wrote. The parts marked `manually graded` will be graded by the course staff after the due date. In addition, we may give you feedback on any part of the code, including automatically graded parts.